

195 PTAS.
(IVA Incluido)
116
mi computer
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
P.V.P. Canarias, Ceuta y Melilla 185 Ptas.
Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen X-Fascículo 116

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones, Francisco Martín
Asesor técnico: Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Aribau, 185, 1.°, 08021 Barcelona
Tel. (93) 209 80 22 - Télex: 93392 EPPA

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 120 fascículos de aparición semanal, encuadernables en diez volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-183-6 (tomo 10)
84-85822-82-X (obra completa)
Depósito Legal: B. 52-84

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 088604
Impreso en España-Printed in Spain- Abril 1986

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (120 fascículos más las tapas, guardas y transferibles para la confección de los 10 volúmenes) son las siguientes:

a) Un pago único anticipado de 27 105 ptas. o bien 10 pagos trimestrales anticipados y consecutivos de 2 711 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S. A. (Aribau, 185, 1.°, 08021 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

No se efectúan envíos contra reembolso.





Marcus Wilson-Smith

Punto de partida
Para quienes no tengan experiencia y deseen hacer una carrera en el ámbito de las ventas o el marketing, probablemente el lugar más adecuado para iniciarse sea una tienda minorista como la que vemos en la fotografía, donde se venden equipos a usuarios finales. No obstante, las perspectivas continúan siendo escasas en un entorno sometido a demasiadas presiones como para permitir que el empleado adquiera una conciencia más cabal y profunda de la nueva tecnología y las técnicas de venta

La senda del éxito

En un mercado tan complejo como el informático, los departamentos de ventas y marketing tienen gran relevancia

El gran crecimiento que experimentó la industria del microordenador a comienzos de los años ochenta comenzó a menguar y estabilizarse hacia mediados de 1984 y, por consiguiente, las expectativas de creación de puestos de trabajo también se redujeron. Se cuentan por centenares las firmas que se han retirado del negocio, pero se han creado casi el triple de ellas, lo que indica que en el campo de la informática se está creando un alto número de colocaciones.

En Gran Bretaña los puestos de trabajo en ventas y marketing figuran entre los mejor pagados en la industria del ordenador. En el extremo superior, el vendedor experimentado que gestiona valiosos contratos de hardware o software con grandes clientes corporativos puede esperar hacerse con £80 000 (17 millones de pesetas, aproximadamente) o más al año, principalmente a base de comisiones. Los ejecutivos de marketing (que no cobran sobre la base de comisiones) tienen pocas posibilidades de ganar tanto dinero, aunque cuando un empleado llega al puesto de director de marketing de una empresa entre mediana o grande son bastante corrientes los sueldos de más de £30 000. Tanto ventas como marketing constituyen sendas convencionales a través de las cuales los empleados más emprendedores y ambiciosos pueden acceder a la cúpula de una empresa. Hasta hace poco tiempo, la totalidad de los principales ejecutivos de IBM habían ascendido a través del escalafón de ventas.

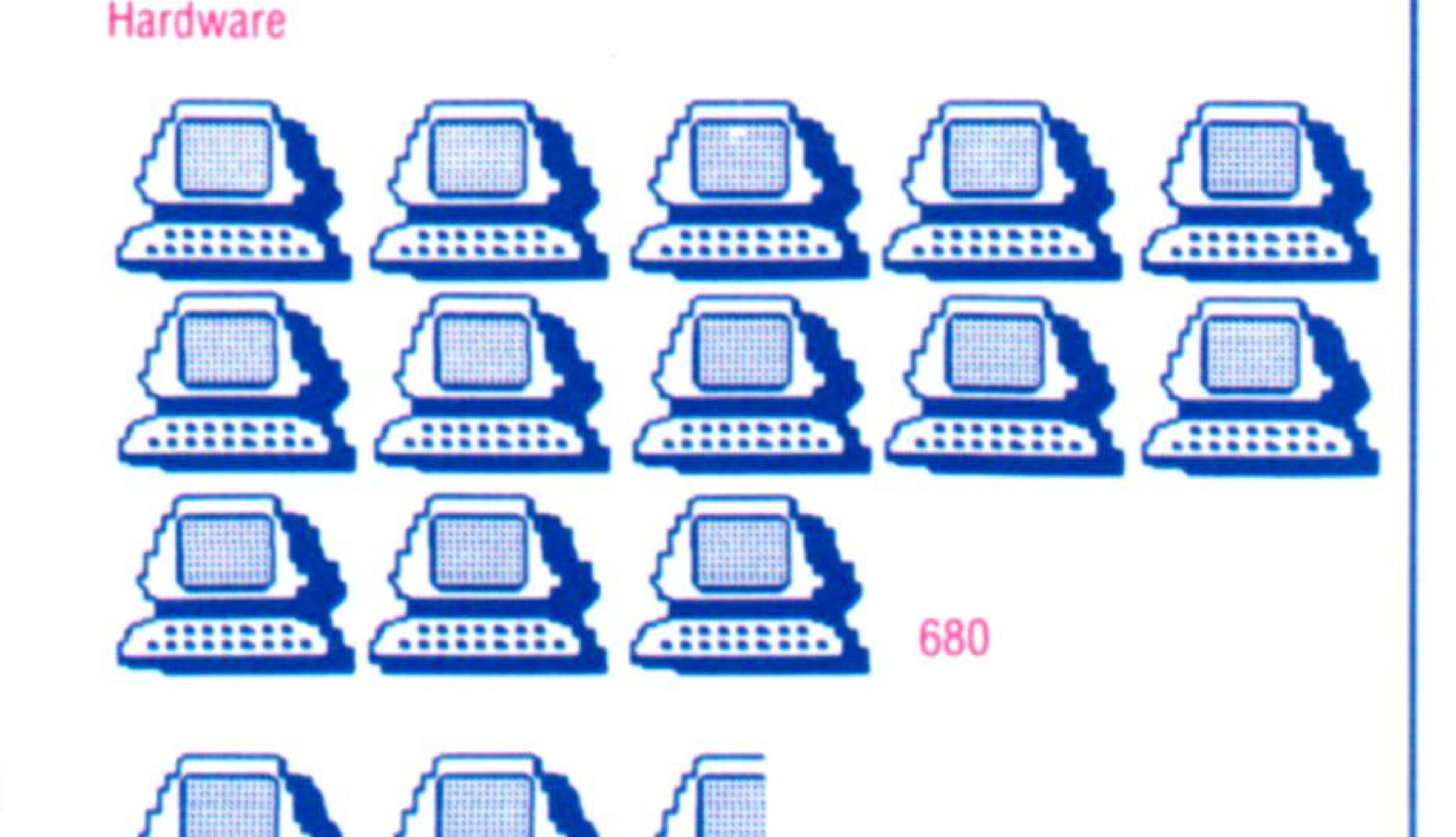
Algunas agencias de colocaciones recomiendan que los jóvenes aspirantes a vendedores empiecen por seguir un curso de informática, con el objeto de obtener una comprensión elemental del producto. Luego aceptar un empleo en una firma minorista que venda equipos de oficina u ordenadores personales. Esto se suele pagar bastante mal, pero les proporcionará alguna experiencia básica. Después de ello, estarán en una posición más ventajosa para convencer a una compañía de fabricación o a un distribuidor de ordenadores para que les dé empleo. El primer año tampoco ganarán mucho, y se esperará que acompañen a un vendedor senior y aprendan a hacer demostraciones de ordenadores y paquetes de software.

Después de esto, si el aspirante demuestra aptitud, sus perspectivas mejorarán rápidamente. Avalado por un año de experiencia en cualquiera de los micros y paquetes de software principales, estará capacitado para un puesto ejecutivo de ventas, cuyo sueldo básico es posible duplicar con las comisiones. Es muy probable que la empresa también ponga a su disposición un automóvil. Cuando se anuncia un trabajo de "£20 000 OTE" (*on-target earnings*), significa que las £20 000 son las "ganancias sobre cuota" que el vendedor percibirá sólo si alcanza el objetivo de ventas que ha establecido para él el director de ventas. (Según los especialis-

Crecimiento de los suministros

Hardware

Cantidad de proveedores en junio de 1985



680

Entradas desde enero 1985

130



40 Retiradas desde enero de 1985

Software

Cantidad de proveedores en junio de 1985

1100



Entradas desde enero 1985

150



70 Retiradas desde enero de 1985

Caroline Clayton

Suministros informáticos
A pesar de las pesimistas previsiones de algunos observadores, la cantidad de proveedores dentro de la industria del ordenador ha seguido creciendo a ritmo constante. El crecimiento estable, más que unos beneficios elevados, a menudo será una tentación para que se creen nuevos negocios en un mercado que, de no existir esta circunstancia, sería evitado (cifras del NCC Microsystems Centre, Gran Bretaña)

El juego de vender

En una empresa las funciones de ventas y marketing con frecuencia se confunden. Un vendedor trata directamente con el cliente. El ejecutivo de marketing, por el contrario, está más implicado en la estrategia: cómo empaquetar el producto, qué nombre ponerle, cómo anunciarlo, etc. En la industria del microordenador hay dos ramas fundamentales: la fabricación y la reventa. La gente de ventas que trabaja para fabricantes de hardware y software típicamente vende sus productos a minoristas o distribuidores. Los vendedores para revendedores, tales como casas de sistemas, detallistas o cadenas minoristas, tratan directamente con el usuario final. En Estados Unidos es corriente que los clientes entren en una tienda de ordenadores y adquieran un ordenador de gestión. Por el contrario, en Gran Bretaña casi todas las ventas de micros implican el envío de correspondencia a los potenciales clientes (una función de marketing) o bien una "llamada en frío" (el vendedor establece un contacto directo por teléfono o personalmente). En los primeros días de la microinformática, la industria no le prestaba gran atención al marketing. Las nuevas empresas fabricaban novedosos y atractivos productos y los vendedores los vendían. En algunas ocasiones tenían éxito (el Spectrum) y en otras no (el Coleco Adam). Ahora, sin embargo, la industria es más sofisticada y en 1985 el fabricante de ordenadores personales de mayor éxito fue Amstrad, con su serie CPC. No había nada nuevo en la tecnología; simplemente fue cuestión de empaquetarla de la forma que querían los clientes. Y eso es, en esencia, lo que se entiende por marketing

tas en gestión de empleo, la mayoría de las empresas muestran preferencia por aspirantes de edades entre veinte y veinticinco años.)

Al cabo de tres o cuatro años, el aspirante puede llegar a ser ejecutivo de ventas senior o vendedor de sistemas. En una empresa que venda ordenadores tanto individuales como multiusuario, el personal senior, evidentemente, manejará los sistemas multiusuario más costosos, que tienen las comisiones netas más elevadas. Para cuando cumpla los treinta años de edad, sus expectativas estarán en ocupar un puesto de director de ventas, a quien corresponderá determinar las cuotas a cubrir por su personal de ventas. A este nivel quizá descubra que sus atribuciones y su campo de acción coinciden en parte con los del director de marketing, y de hecho muchos ejecutivos en este momento deciden pasarse a esta última actividad. Tenga presente que las ventas pueden ser un asunto comprometido, especialmente en la industria del micro. Muchos vendedores se han encontrado con que su director de ventas les asignaba objetivos inalcanzables. Al no conseguir acceder a la meta fijada durante dos o tres trimestres, fueron despedidos. ¡Éste no es un trabajo para aquellos que busquen seguridad!

Si descubre que no le agradan las ventas, o que no es especialmente idóneo para vender, el mejor consejo es abandonar el negocio mientras aún sea joven. Si mediada la treintena aún no ha accedido a un puesto de director de ventas, quizá le resulte difícil conseguir otro empleo en este campo.

El marketing es un ámbito muy competitivo. Las firmas buscan graduados universitarios, preferentemente con titulación en matemáticas o física.

Un punto importante que hay que entender en relación al marketing es que rigen los mismos principios independientemente del producto que venda la compañía comercial. De modo que la experiencia obtenida en el departamento de marketing de una organización que fabrique jabón en polvo coloca al aspirante en una buena posición cuando llegue al mercado de ordenadores. De hecho, muchas agencias de empleo recomiendan que los graduados jóvenes trabajen durante dos o tres años en una gran corporación, preferiblemente alguna que los haga asistir a un curso de marketing o que les permita obtener un título en administración de empresas. Las firmas de ordenadores, al ser más pequeñas, suelen contar con menos recursos para formar profesionalmente a sus empleados.

El conocimiento en materia de ordenadores no es vital para obtener un puesto de marketing en una firma de microordenadores, pero es muy conveniente interesarse en el tema. Un empleado de marketing con dos o tres años de experiencia ganará unas £10 000 al año. Las tareas que deberá realizar para la firma incluirán escribir folletos y ejemplares publicitarios, enviar información a la red de ventas, organizar seminarios y conferencias de prensa, y atender a los periodistas (normalmente en esta última actividad será sustituido por alguien más especializado en este campo específico).



Reacción en cadena

La firma británica Granada Business Centres vende micros individuales y multiusuario a pequeñas y medianas empresas. La empresa declara que el requisito primordial de sus aspirantes a un puesto de vendedor es que posean experiencia en ventas, preferiblemente de equipos de oficina o, aun mejor, de sistemas de ordenador. Granada imparte su propia formación en productos y aptitudes para las ventas, estando comprendidas las edades del personal de ventas que contrata entre los 23 y 30 años. Cada tienda posee un ejecutivo de ventas senior que se ocupa de las grandes "cuentas nacionales", y un gerente de ventas que supervisa la actividad de ventas de su tienda. Después hay un gerente de distrito que se ocupa de un grupo de cuatro tiendas incluyendo al personal de mantenimiento y apoyo que trabaja para cada centro. El camino para hacer carrera es éste:

vendedor→ejecutivo de ventas senior→ gerente de ventas→gerente de distrito

El personal de marketing es mucho más reducido: cuatro personas, frente a 60 en el campo de ventas. Cada gerente de marketing suele cumplir una función específica. Una persona se encarga de las ferias, otra de la prensa y relaciones públicas, otra de la presentación de las tiendas, etc.

El siguiente paso conduce al cargo de gerente de marketing, donde se asume un papel más directo en establecer enlaces con las agencias de publicidad, decidir a qué ferias presentarse, planificar lanzamientos de productos y la estrategia de la empresa. El sueldo base sería de unas £14 000, con un coche de la empresa y otras prerrogativas. En una empresa pequeña (como son casi todas las firmas de microordenadores) sería razonable esperar convertirse pronto en director de marketing con un asiento propio en la mesa de juntas.

Pequeño ejemplo

Psion es una firma británica que produce software y ordenadores de mano. Tiene contratadas a 70 personas y espera que sus aspirantes a un puesto de ventas posean algo de experiencia en cuanto a la venta de equipos técnicos o de oficina, pero no hace especial hincapié en el conocimiento sobre ordenadores. Ofrece formación sobre los productos de la compañía. El vendedor tratará directamente con los comerciantes que llevan los productos de Psion y los grandes compradores corporativos.
Un camino típico para hacer carrera sería:

vendedor→gerente de cuentas→gerente de ventas→director de ventas

Un gerente de cuentas es responsable de manejar varias cuentas y supervisar la capacidad de ventas. En el campo del marketing, Psion sólo contrata graduados y ejecutivos de marketing. Su tarea consiste en escribir los textos publicitarios, organizar los stands de las ferias, atender a la prensa y proporcionar apoyo al cuerpo de ventas. Aquí el camino sería:

ejecutivo de marketing→gerente de marketing→director de marketing

El gerente y el director de marketing son responsables de la planificación a largo plazo, de tratar con las agencias de publicidad, de encargar estudios de mercado, etc.
Las pequeñas empresas pueden ofrecer ventajas a un empleado que no esté seguro del camino a seguir, dado que su estructura es más flexible que la de las grandes firmas. En 1983, Psion contrató como programador a un graduado que luego pasó al apoyo técnico de ventas (respondiendo a cuestiones que el equipo de ventas no podía manejar solo) antes de ascender a gerente de cuentas de ventas. Hacia finales de 1985 había pasado a la división de exportaciones



Cortesía de Amstrad

Acierto publicitario
El marketing representa muchísimo más que vender un producto. Un ejecutivo de marketing estará comprometido con un producto desde la etapa de concepción y diseño hasta su lanzamiento, asegurándose de que durante el desarrollo del producto se tengan muy presentes las exigencias y preferencias del usuario final. El Amstrad PCW 8256 constituye un buen ejemplo de producto bien comercializado. Si bien utiliza la tecnología de ayer, responde a las necesidades del consumidor de hoy en cuanto a tratamiento de textos a bajo costo. (El slogan publicitario reza: "Más que un procesador de textos, por menos que una máquina de escribir".) Compare su éxito con, por ejemplo, el vehículo eléctrico Sinclair C5, que, a pesar de la gran campaña publicitaria que acompañó su lanzamiento, no logró venderse bien, lo que demuestra que la publicidad, por intensiva que sea, no puede vender un producto para el cual no existe un mercado

¡Utilidades Unix!

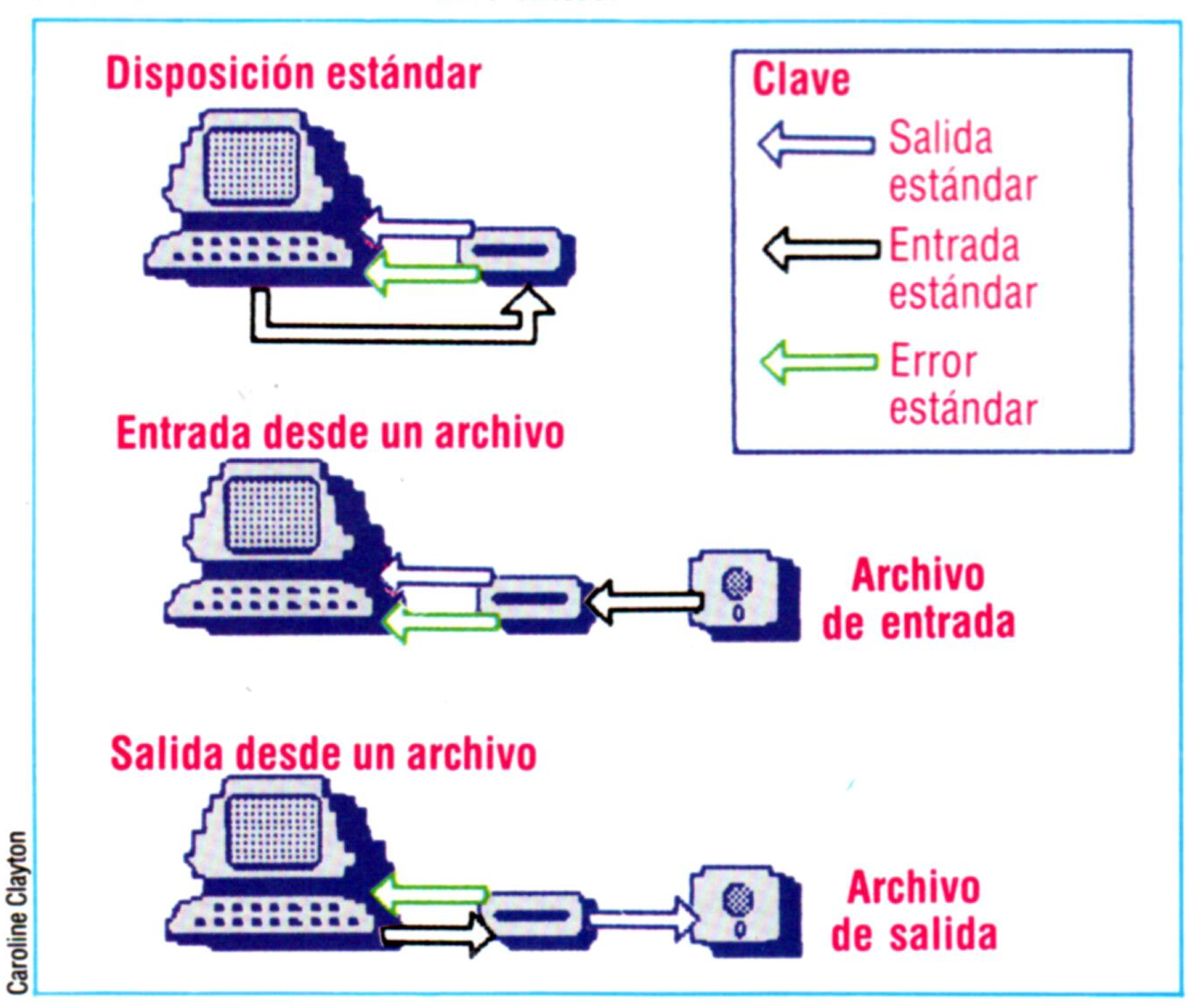
Unix ofrece una biblioteca de llamadas y comandos para simplificar la labor del usuario

Debido a que las utilidades y herramientas Unix son programas en c directamente, es fácil combinar con ellas los programas de uno mismo. Para simplificar aún más las cosas, el Unix tiene un enfoque unificado fundamental de los archivos y de dispositivos de E/S. Se considera que cada archivo es una corriente o secuencia de bytes. El Unix lee o escribe secuencialmente los bytes en el archivo y permite posicionar el puntero de archivo en cualquier desplazamiento de byte legítimo del archivo y comenzar a leer o escribir en este punto.

Los dispositivos de E/S se tratan exactamente igual que cualquier otro archivo. De hecho, en el directorio /dev siempre habrá una lista de dispositivos de E/S. Las operaciones que se pueden efectuar se observan mejor utilizando las llamadas a la biblioteca c de nivel más bajo:

- creat(nombrearchivo,modalidadprotección): Si ya existe un archivo con el nombre dado, entonces se lo trunca a longitud cero; de lo contrario, se crea un archivo con ese nombre. La modalidadprotección es un número de nueve bits (comúnmente en forma de tres dígitos octales) que especifican la permisión de lectura, escritura y ejecución (un bit cada uno) para el usuario, el grupo del usuario y de quien esté en el sistema. La función devuelve un descriptorarchivo entero.
- open(nombrearchivo,modalidadlecturaescritura): Conecta el archivo mencionado con el programa, especificando (en modalidadlecturaescritura) 0 para lectura, 1 para escritura y 2 para acceso tanto de lectura como de escritura. El archivo debe existir ya y la función devuelve un descriptorarchivo entero.
- close(nombrearchivo): Desconecta el archivo del programa.
- unlink(nombrearchivo): Suprime el archivo del sistema.
- read(descriptorarchivo,buffer,cantidaddebytes) y write(descriptorarchivo,buffer,cantidaddebytes): Transfieren el número especificado de bytes entre el archivo, del cual se da el descriptor, y una zona de almacenamiento de buffer mencionada.

Rutas de redirección
En circunstancias normales, la entrada para un programa que se esté ejecutando en un sistema Unix se produce a través del teclado. La salida y los avisos de error normalmente se dirigen hacia la pantalla. En el Unix, los operadores < y > permiten redirigir la entrada a un programa o bien la salida del mismo, de modo que el programa se comunique con otros dispositivos o archivos. Esta potente facilidad permite crear con toda sencillez archivos de datos, o bien que el "caparazón" ejecute una serie de comandos Unix retenidos en un archivo



Caroline Clayton

Tres archivos

Hay tres archivos abiertos automáticamente para cualquier programa, que se denominan standardinput, standardoutput y standarderror. Normalmente, standardinput está conectado al teclado y standardoutput y standarderror están conectados a la pantalla. Los dos primeros se pueden redirigir hacia o desde cualquier otro archivo o dispositivo mediante los operadores < y >.

Para cualquier comando o programa Unix:

```
nombreinstruccion<nombrearchivoodispositivo
```

hará que la entrada desde standardinput se tome realmente del archivo mencionado en vez de desde el teclado.

Del mismo modo:

```
nombreinstruccion>nombrearchivoodispositivo
```

hará que la salida de standardoutput se envíe al archivo mencionado en vez de a la pantalla. Una variación permitirá añadir la salida a un archivo en lugar de crear un archivo nuevo, si bien se creará un archivo nuevo si éste no existe:

```
nombreinstruccion>>nombrearchivoodispositivo
```

Las acciones de estas dos últimas se pueden alterar si se establece noclobber (una opción especial que más adelante veremos en mayor profundidad). Ésta no permitirá que se supriman archivos sin confirmación, pero se puede invalidar con la forma imperativa de estos dos operadores:

```
nombreinstruccion>!nombrearchivoodispositivo
nombreinstruccion>>!nombrearchivoodispositivo
```

A modo de ejemplo del empleo de esta facilidad, podemos crear un archivo que contenga la lista de archivos de un directorio utilizando:

```
ls>lsarchivo
```

otro uso es para proporcionar una facilidad de archivo de comandos, de modo que toda una secuencia de comandos se pueda ejecutar a la vez. Los comandos los ejecuta el "caparazón", el cual se puede invocar utilizando la instrucción sh. De modo que:

```
sh<nombrearchivo
```

hará que se ejecuten los comandos del archivo.

Una característica fundamental del Unix, que le confiere tanta potencia, es la facilidad de tubería que da el operador |.

Responde al formato:

nombrecomandoprograma|nombrecomandoprograma

que conecta standardoutput del primer comando con standardinput para el segundo. De modo que la salida del primer comando se toma automáticamente como la entrada para el segundo.

Esto no significa que se produzca un archivo intermedio, sino que los dos programas se ejecutarán de forma concurrente. Siempre que el segundo programa necesite entrada, suspenderá la operación hasta que el primer programa haya producido alguna salida. Por ejemplo, la instrucción ls no cuenta la cantidad de archivos del directorio mientras los va listando.

Existe una utilidad, wc, que cuenta la cantidad de palabras de un archivo utilizando la opción –w; el comando:

ls | wc–w

hace que la salida de ls (una lista de nombres de archivo) sea la entrada para wc, que cuenta el número de palabras, de modo que la salida será la cantidad de archivos del directorio.

El operador de tubería tiene un uso especial en conjunción con el comando tee, que enviará salida tanto a un archivo como a la pantalla. Así:

ls | tee lsarchivo

hará que el listado del directorio aparezca en la pantalla así como en lsarchivo.

El Unix es un sistema operativo multitarea y, en consecuencia, puede ejecutar dos o más tareas de forma concurrente. Pero también permite que cada usuario ejecute más de un programa a la vez. Si se coloca el operador & al final de la línea de comando, el comando o programa se ejecutará como tarea de fondo, dejando al usuario en libertad de ejecutar otro comando o programa. Por supuesto, si este proceso de fondo utiliza una entrada o salida de terminal, entonces es probable que ocurran cosas extrañas.

También se puede detener una tarea en medio de su ejecución pulsando Ctrl-Z, que permite reiniciar la tarea después, sin perderla. Si el usuario está ejecutando simultáneamente varias tareas y algunas se interrumpen, puede resultar difícil saber lo que está sucediendo. No obstante, el comando ps da una lista de los estados de todos los procesos que se están ejecutando para el usuario, y el comando jobs lista todas las tareas de fondo y las que están detenidas.

Cuando se detiene un proceso o se lo coloca como tarea de fondo, el Unix le otorga dos números: un número de tarea personal, como [1], [2], etcétera, y un número de proceso que está en relación con todos los procesos que se estén ejecutando actualmente en la máquina. El número de tarea se puede utilizar en unión con los comandos fg y bg para intercambiar procesos. El comando fg%númerodetarea, por ejemplo, reiniciará un proceso detenido en el primer plano, o pasará un proceso de fondo a primer plano. El comando bg%númerodetarea reiniciará un proceso de fondo, y mediante stop%númerodetarea se puede detener un proceso que se esté ejecutando en el fondo. Los procesos que ya no son necesarios se pueden detener permanentemente mediante el comando kill númerodeproceso.

Dirigiendo el tráfico

```
%ls –l > lsfile
(no aparece el listado del directorio en la pantalla)

%cat lsfile           (mirar en lsarchivo)
total 41
–rw–r—r—      1 com–mcc      0  Oct 28  11:55 lsfile
drwxr–xr–x    2 com–mcc    512  Oct 21  11:11 mike
–rw–rw–r—     1 com–mcc    502  Sep 17  12:07 rec.c
–rwxr–xr–x    1 com–mcc  18432  Oct 21  11:02 receive
–rw–r—r—      1 com–mcc   1068  Oct 18  14:44 rx.p
–rwxr–xr–x    1 com–mcc  19456  Oct 21  11:03 transmit

%ls | sort –r > lsfile
(esta vez la lista del directorio se ha entubado a la utilidad sort con la opción -r, que clasifica por orden inverso)

%cat lsfile     (listar archivo clasificado)
transmit
rx.p
receive
rec.c
mike
lsfile

%pc rx.p &      (ejecutar compilación pascal como tarea de fondo)
[1] 1217        (se le da a la compilación el n.° de tarea [1], y al proceso el n.° 1217)

%jobs
[1] + Running pc rx.p

%ps
PID   TT     STAT TIME COMMAND
1217  n09 I  0:00  pc rx.p
1218  n09 R  0:03  pc0 -o/tmp/p0001217 rx.p
(this last process was set up by the pascal compiler)
1220  n09 R  0:01  ps

%stop 1217          (detenida compilación por citar el n.° de proceso)
[1] +Stopped(signal) pc rx.p

%bg %1              (compilación reiniciada en el fondo utilizando el n.° de job)
[1] pc rx.p &

%fg %1              (pasar compilación al primer plano)
pc rx.p
Stopped             (esto se hizo utilizando Ctrl Z)

%jobs
[1] +Stopped pc rx.p

%kill %1            (librarse por completo del proceso)
[1] Exit 1 pc rx.p

%ps
PID   TT   STAT TIME  COMMAND
1266  n09  R    0:01  ps
(ningún proceso en ejecución, a excepción de ps)

%logout
```

El Unix, como todos los sistemas operativos buenos, mantiene una hora y fecha que se pueden utilizar de diversas maneras. La instrucción date simplemente proporciona la fecha del día y cal un calendario, que puede presentarse con diversos formatos. La instrucción times nombrecomando le dirá la hora de ejecución para ese comando en particular, y at seguida por una hora y un nombre de archivo ejecutará una lista de comandos retenidos en el archivo en la hora especificada.

La hora del modem

Si bien el modem fue creado hace varios años, sólo ahora se ha generalizado su utilización

En Estados Unidos los modems constituyen un gran negocio. En Europa, sin embargo, su impacto ha sido menos pronunciado. Esto se debe a diversos motivos, que pueden reducirse a dos factores principales. En primer lugar, los recargos por conexión a la red telefónica, en particular durante las horas punta, pueden resultar prohibitivamente altos. Las personas que usan sus ordenadores en la red durante varias horas por semana con frecuencia se quedan atónitas ante su factura telefónica, que puede ascender a muchos miles de pesetas.

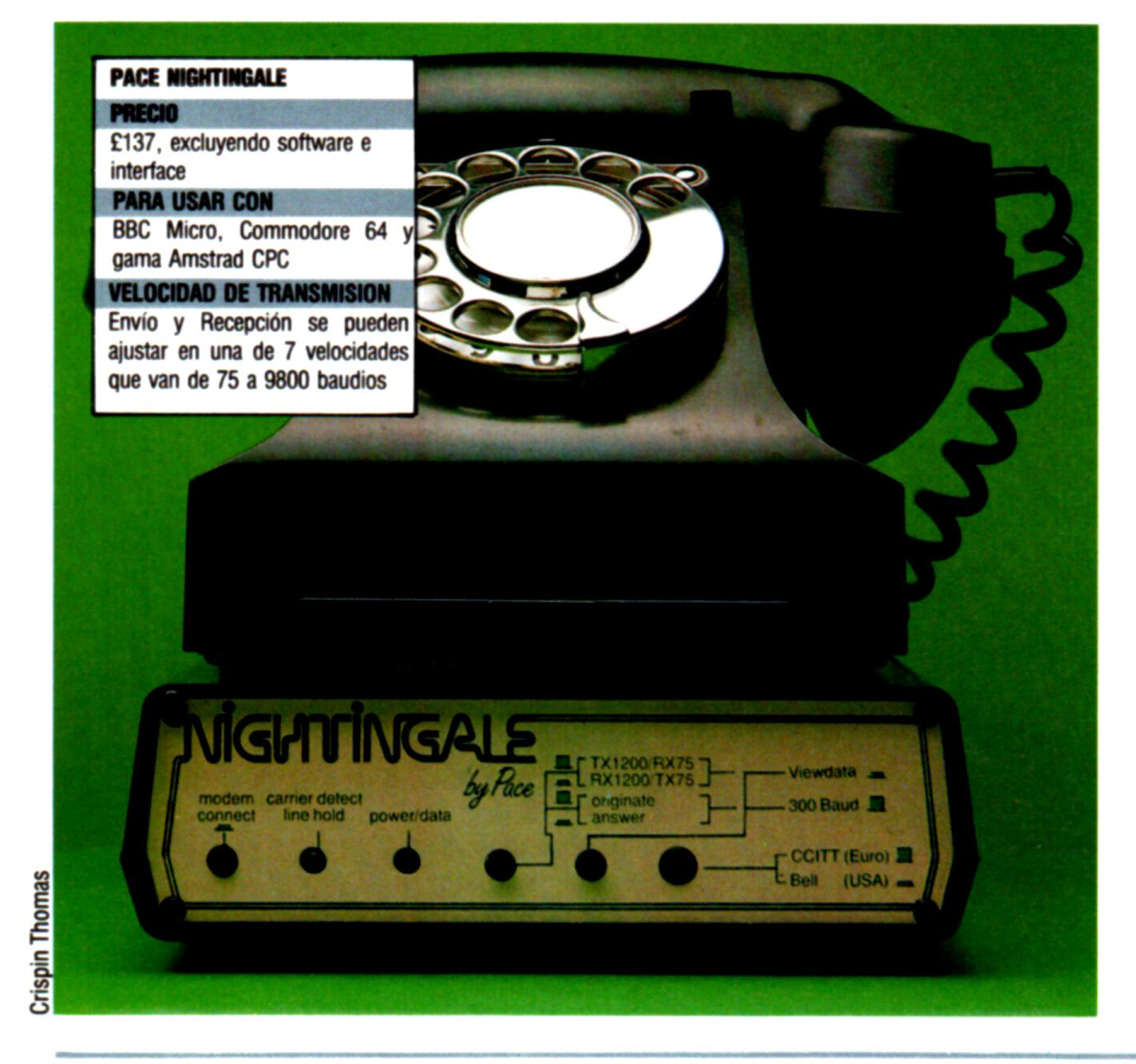
En segundo lugar, las compañías telefónicas han tardado bastante en comprender el potencial de las redes de comunicaciones y han desarrollado su sistema de un modo fragmentario. Esto significa que la base de datos Prestel, por ejemplo, utiliza distintos protocolos de tablones de anuncios privados, etcétera.

Otro factor que ha obstaculizado el desarrollo en los países europeos es el hecho de que antes de que se pueda utilizar un modem legalmente en la red telefónica, normalmente debe ser homologado por la compañía telefónica de turno, y la lentitud de estas empresas para conceder esta aprobación siempre ha sido notable. Ello ha significado que en muchos casos transcurrieran años antes de que los avances en la tecnología del modem llegaran hasta el usuario. En el caso de Gran Bretaña, desde la privatización de la BT (British Telecom), la autorización de los modems se ha transferido al Departamento de Comercio e Industria, lo que habría de volver más eficaz el proceso.

En Estados Unidos, muchas llamadas locales son gratuitas y casi todas las bases de datos y sistemas de comunicaciones se basan en el estándar Hayes. Éste es un sistema de protocolos que ha sido adoptado por otros numerosos fabricantes de modems. Esta normalización no existe para el usuario británico, aunque el desarrollo del sistema Hayes en este país es inminente.

En consecuencia, para ser totalmente adaptable en Gran Bretaña, es imprescindible que el modem tenga incorporadas estas y otras numerosas características para permitir al usuario la máxima flexibilidad en la a menudo espesa maraña de las comunicaciones.

A pesar de estos problemas, la venta de modems continúa aumentando. Veamos ahora las facilidades que ofrecen algunos de los modems disponibles en el mercado británico.



PACE NIGHTINGALE

PRECIO
£137, excluyendo software e interface

PARA USAR CON
BBC Micro, Commodore 64 y gama Amstrad CPC

VELOCIDAD DE TRANSMISION
Envío y Recepción se pueden ajustar en una de 7 velocidades que van de 75 a 9800 baudios

Crispin Thomas

Nightingale

El Pace Nightingale está dirigido específicamente al BBC Micro, si bien puede ser adoptado por cualquier micro que posea una puerta RS232. Al igual que el paquete Modem 1000, proporciona el software en una EPROM que se inserta en uno de los conectores de ROM laterales del BBC Micro. Si bien varias de las facilidades del modem están contenidas en el software activado por menú, el Nightingale permite establecer la velocidad de transmisión, a través de botones situados en la parte frontal del dispositivo, según los estándares del Prestel y los tablones de anuncios de 300 baudios más comunes. El software adapta el ordenador para la operación en Prestel o en modalidades de emulación de terminal. Desde cualquiera de estos menús se puede transferir información y manipularla dentro del buffer de la máquina. Asimismo, el Nightingale viene con un manual que es el más exhaustivo de los que acompañan a todos los modems que analizamos en este capítulo. Debido a la vasta cantidad de BBC Micros instalados en establecimientos educativos, probablemente los fabricantes hayan diseñado el manual teniendo presente el mercado educativo. Pero, sea cual fuere la razón, le dice al usuario mucho más de la escasa información que proporcionan otros muchos modems pensados para micros personales.



VTX 5000

A pesar de su historia azarosa, el VTX 5000, fabricado por OE Ltd, se ha convertido en el modem estándar para usuarios del Spectrum. Se desarrolló originalmente para Prism, que estaba interesada en ampliar la base de usuarios para Micronet, en la que la empresa tenía grandes participaciones. Aunque cuando Prism fue a la quiebra, a comienzos de 1985, el modem fue asumido por Modem House, siguió vendiéndose bien.

En la parte frontal del modem hay una luz de potencia, una luz Line (que indica si el modem está "en línea") junto con un interruptor. Otro interruptor permite seleccionar la modalidad requerida para el dispositivo, que puede ser para Micronet, TX (transmitir) o RX (recibir). En la parte posterior del dispositivo hay un conector para enchufe telefónico y un cable, que se pueden enchufar directamente a un conector telefónico de pared.

El software del VTX 5000 está contenido en ROM. En el interior del VTX 5000 hay dos placas de circuito impreso, una que se ocupa de la transmisión en serie y sistema de circuitos de decodificación, mientras que la otra contiene la lógica específica del Spectrum, tales como el chip ACIA de alimentación de potencia y el software basado en ROM. Este método de proporcionar el software del modem presenta un problema al usuario del Spectrum. La ROM trabaja paginando la ROM de BASIC y ocupando esa zona de memoria. El problema se plantea cuando los usuarios desean utilizar la Interface 1 junto con el VTX 5000, para, por ejemplo, cargar programas en un microdrive. Debido a que la Interface 1 aplica la misma técnica para paginar su propia ROM, los dos dispositivos no se pueden utilizar juntos.



El software del modem es activado por menú y tras el encendido presenta un menú de opciones, incluyendo conexión con Micronet o envío de mensajes a través de la facilidad de buzón de correo. Sin embargo, el software no permite conectar con ninguno de la multitud de tablones de anuncios privados que están surgiendo a lo ancho y a lo largo de Europa, si bien es posible escribir software para hacerlo.

VTX 5000
PRECIO
£69,95
PARA USAR CON
Sinclair Spectrum
VELOCIDAD DE TRANSMISION
75/1200 baudios

Modem 1000

Con el mismo origen que el VTX 5000, el Modem 1000 tiene muchas facilidades en común con la máquina exclusiva para el Spectrum. Sin embargo, el Modem 1000 está diseñado para operar con una amplia variedad de ordenadores y, en consecuencia, no lleva ninguna ROM específica en su placa.

Los controles externos son idénticos a los suministrados en el VTX 5000. De modo que hay un interruptor Line y un LED, y un interruptor de modalidad de tres vías. La parte trasera del dispositivo contiene una puerta Data In que proporciona la interface para el ordenador, así como un enchufe telefónico. Puesto que el Modem 1000 no se alimenta desde el ordenador, se ha dotado al dispositivo de su propio cable y transformador de potencia.

El sistema de circuitos de transmisión en serie y chip ACIA es casi idéntico al utilizado en el VTX 5000. Sin embargo, para que el usuario se comunique a través del sistema se requiere algo de software. Para el BBC Micro, éste se proporciona en ROM, que se puede instalar en uno de los conectores de ROM laterales libres dentro del ordenador. Esto proporciona facilidades muy similares a las del Spectrum. Programas activados por menú permiten cargar programas, conectar con Micronet y recibir y transmitir mensajes.

Por la propia naturaleza del Commodore 64, la interface requerida para el Modem 1000 es más compleja que aquella para el VTX 5000. La misma se proporciona mediante una placa de interface conectable que se instala en la puerta para cartuchos del ordenador. Esta interface no sólo contiene el software que se paginará en la memoria en el lugar de la ROM de BASIC del ordenador, sino también un sistema de circuitos adaptador especial que programa el conjunto ASCII Commodore según el estándar de comunicaciones utilizado generalmente. Otra parte de la placa produce una señal RS232 estándar que se le puede enviar a la puerta de datos del modem.



MODEM 1000
PRECIO
£69,95, excluyendo software e interface
PARA USAR CON
Todos los micros personales
VELOCIDAD DE TRANSMISION
75/1200 baudios



Communicator 104

Fabricado por KDS Electronics, este modem está diseñado para usar sólo con la gama de máquinas Amstrad CPC. Al igual que VTX 5000, se enchufa directamente en el bus de ampliación de la parte posterior del ordenador. Pero en este caso la potencia se suministra a través de un transformador externo instalado en el conector de potencia. Asimismo, al igual que los otros modems que hemos analizado, éste se conecta en serie con el teléfono. En consecuencia, para ponerse en línea con un tablón de anuncios, se debe primero marcar el número en el teléfono y después encender el modem.

El Communicator se diferencia de los otros modems de que hablamos aquí en que posee en el frente dos visualizaciones en siete segmentos que le proporcionan información tal como el número que se está discando actualmente, y si la ROM del modem está conectada en el sistema Amstrad.

El software para el Communicator está instalado en ROM dentro del modem propiamente dicho. Al igual que la gama OEL, los programas son activados por menú, si bien la cantidad de opciones disponibles es muchísimo mayor. El menú Mode Select (selección de modalidad) le permite seleccionar el tipo de sistema que desea utilizar: Prestel o tablón de anuncios, por ejemplo. Una vez seleccionada la modalidad, el modem ajustará automáticamente la velocidad de transmisión, si bien usted puede elegirla por sí mismo. El software activado por menú también soporta facilidades para llamada y respuesta automática y detección de portadora.

El Communicator soporta una vasta cantidad de útiles comandos residentes de ampliación del sistema. Éstos incluyen, entre otras cosas, comandos para entrar la ROM del modem, establecer la velocidad de transmisión y opciones de palabra de datos, y configurar la impresora para que proporcione una salida impresa de la sesión. El principal inconveniente del modem es que no se puede conectar el teléfono normal mientras se está utilizando. Por tanto, es necesario enchufar el modem o teléfono en el conector de pared con el fin de seleccionar la pieza del equipo que desea utilizar.



Crispin Thomas

WS3000

Comercializado por Miracle Technology, esta gama de modems se halla en el extremo superior del espectro de microordenadores, conteniendo la gama completa de facilidades que requieren la mayoría de los usuarios.

Hay tres modelos de la gama WS3000 que cubren una amplia variedad de velocidades de transmisión y protocolos, desde 75/1200 bps (bits por segundo) half duplex (para acceder al Prestel) hasta 1200/1200 bps transmisión full duplex para transmisiones de usuario a usuario. En el frente del modem hay varias luces indicadoras, que informan al usuario del estado actual del dispositivo. Éstas incluyen la operación de las patillas en la puerta RS232 y si el terminal está o no en línea. Cada uno de los tres modelos contiene utilidades que soportan llamada automática, respuesta automática y una puerta RS232 con buffer.

Además, la gama WS3000 soporta el sistema de protocolos de estándar Hayes. Éste es particularmente útil para los usuarios de gestión, dado que es creciente la cantidad de paquetes de gestión que incluyen facilidades que permiten enviar datos a través de un modem directamente desde el programa (la mayor parte de estos paquetes emplean protocolos Hayes).

Alien

Usted es el comandante de una nave de combate y se interna en una galaxia desconocida... Es el comienzo de una gran aventura a los mandos de su ordenador Atari

Su misión es destruir a los enemigos que le rodean y le impiden aterrizar. La única manera de neutralizar al enemigo es destruir todos los espacios que lo rodean.

Usted solamente puede evolucionar dentro del campo de batalla. Los enclaves prohibidos están representados por signos "–". Si aterriza directamente sobre el enemigo (representado por la letra A), usted será destruido. El adversario se puede desplazar una o dos casillas a la vez, pero puede también permanecer inmóvil. Usted debe impartir sus instrucciones en la forma de dos coordenadas. No olvide pulsar la tecla Return después de cada una de ellas.

Cada posición destruida aparece como una casilla en blanco; las otras están representadas por los signos "*".

El enemigo observa el desarrollo de la batalla y le informa del grado de peligro que percibe. La mejor manera de combatirlo es lograr bloquearlo en un rincón, porque esto limita considerablemente sus posibilidades de desplazamiento. El adversario recela de esta estrategia y, utilizando la "inteligencia" en las líneas 6145 y 6161, intentará evitar los cuatro rincones del campo de batalla. Trate de rodear al enemigo de la forma más rápida posible con el fin de limitar su campo de acción. ¡Hay un récord que usted puede batir!

```
----DEGRE DE DANGER SENTI PAR L'ENNEMI
-4---
DEROULEMENT DE LA BATAILLE:
  LE RECORD EST DE 0
------------------------------------

 POSITION DE L'ENNEMI 9 4

TEMPS: 24    TIRS EFFECTUES: 6

12345678910
----------1
-********-2
-****** *-3
-***   *A-4
-******  -5
-********-6
-********-7
-********-8
-********-9
----------10

COORDONNEES DE VOTRE TIR
HAUT?
```

```
1 GRAPHICS 0:SETCOLOR 4,8,0:SETCOLOR 2,8,0
3 DIM A(10,10)
5 HISCORE=0
10 REM ALIEN
20 REM (C)HARTNELL 1982
30 GOSUB 9000:REM INICIALIZACION
40 GOSUB 8000:REM LETREROS
50 GOSUB 7000:REM MOVIMIENTO JUGADOR
60 GOSUB 6000:REM MODIF ENEMIGO
70 TEMPS=TEMPS-1
75 TIRS=TIRS+1
80 IF TEMPS=0 THEN 6570
85 GOSUB 8000
90 GOTO 50
910 TEMPS=30
5000 REM COLISION
5010 PRINT "HA TOCADO AL CAPITAN ENEMIGO"
5020 PRINT "Y LO HA DESTRUIDO"
5030 GOTO 6570
6000 REM MODIF ENNEMI
6010 REM TEST SI ENCERCLE
6020 H=0
6030 IF A(M-1,N)=2 THEN H=H+1
6040 IF A(M-1,N-1)=2 THEN H=H+1
6050 IF A(M,N-1)=2 THEN H=H+1
6060 IF A(M,N+1)=2 THEN H=H+1
6070 IF A(M-1,N+1)=2 THEN H=H+1
6080 IF A(M+1,N+1)=2 THEN H=H+1
6090 IF A(M+1,N-1)=2 THEN H=H+1
6100 IF A(M+1,N)=2 THEN H=H+1
6110 IF H=8 THEN 6500:REM CERCO
6120 REM DESPLAZAMIENTO DEL ENEMIGO
6125 E=M:F=N
6130 M=M-INT(RND(1)*2)+INT(RND)1)*2)
6140 IF M<2 OR M>9 THEN 6130
6145 IF (M<4 OR M>7) AND RND(1)>0.7 THEN 6130
6150 N=N-INT(RND(1)*2)+INT(RND(1)*2)
6160 IF N<2 OR N>9 THEN 6150
6161 IF (N<4 OR N>7) AND RND(1)>0.7 THEN 6150
6162 IF A(M,N)=2 THEN 6130
6165 A(E,F)=0
6170 A(M,N)=1
6300 RETURN
6500 REM CERCO
6505 GOSUB 8000
6510 PRINT "LO HA CONSEGUIDO! BRAVO"
6520 PRINT "HA FALLADO "; TIRS; "TIRS"
6530 PRINT "Y LO HA HECHO EN"; TEMPS, "MINUTOS", "LEFT"
6540 Q=TEMPS*125.67
6550 PRINT "SU SCORE       ";Q
6560 IF Q>RECORD THEN RECORD=Q
6570 PRINT "EL RECORD ES ";RECORD
6580 PRINT
6590 PRINT "PULSE 1 PARA VOLVER A JUGAR, 2 PARA
     DETENERSE"
6600 INPUT A
6610 IF A=1 THEN 10
6620 PRINT "A LA PROXIMA CAPITAN"
6630 END
7000 REM MOVIMIENTO JUGADOR
7010 PRINT "COORDENADAS DE SU DISPARO"
7020 PRINT "VERTICAL.";
7030 TRAP 7030:INPUT S:TRAP 40000
7035 IF S<2 OR S>9 THEN 7030
7040 PRINT "HORIZONTAL.";
7050 TRAP 7050:INPUT R:TRAP 40000
7055 IF R<2 OR R>9 THEN 7050
7066 IF A(R,S)=1 THEN 5000:REM DESTRUCCION DEL ENEMIGO,
     FIN DE LA PARTIDA
7070 IF A(R,S)=2 THEN PRINT "SECTOR YA DESTRUIDO":FOR
     P=200 TO 255:SOUND 0,P,10,15:NEXT P:SOUND 0,0,0,0
7090 IF A(R,S)=2 THEN RETURN
7100 A(R,S)=2
7110 RETURN
8000 REM LETREROS
8005 PRINT CHR$(125)
8020 PRINT
8030 "PRINT GRADO DE PELIGRO DETECTADO POR EL
     ENEMIGO-";H;"—"
8050 PRINT "DESARROLLO DE LA BATALLA:",, "EL RECORD ES DE
     "; RECORD
8060 PRINT " ------------------------------------------------
     ----------------"
8080 PRINT
8090 PRINT "POSICION DEL ENEMIGO ";N;" ";M
8100 PRINT
8110 PRINT "TEMPS: ";TEMPS;" DISPAROS EFECTUADOS: ";TIRS
8120 PRINT
8125 PRINT "12345678910"
8130 FOR K=1 TO 10
8140 FOR J=1 TO 10
8145 IF K<2 OR K>9 OR J<2 OR J>9 THEN PRINT "-";
8146 IF K<2 OR K>9 OR J<2 OR J>9 THEN 8180
8150 IF A(K,J)=0 THEN PRINT "*";:SOUND 0,*8+J*4,10,15
8160 IF A(K,J)=1 THEN PRINT "A";:FOR P=20 TO 0 STEP -1:SOUND
     0,P,12,15:NEXT P
8170 IF A(K,J)=2 THEN PRINT " ";:FOR P=4 0 TO 0 STEP -2:SOUND
     0,P,8,15:NEXT P
8180 SOUND 0,0,0,0:NEXT J
8190 PRINT K
8200 NEXT K
8210 PRINT
8990 RETURN
9000 REM INICIALIZACION
9010 TEMPS=30
9020 TIRS=0
9030 H=0
9050 FOR B=1 TO 10
9055 FOR C=1 TO 10
9060 A(B,C)=0
9065 IF B<2 OR B>9 OR C<2 OR C>9 THEN A(B,C)=2
9067 NEXT C
9070 NEXT B
9080 M=INT(RND(1)*7)+2
9090 N=INT(RND(1)*7)+2
9100 A(M,N)=1
9900 RETURN
```



Frédéric Voisin

Dibujo generado por ordenador
Esta salida impresa está tomada del ImageWriter. El dibujo fue encargado por la casa discográfica C.S.A. para ilustrar la cubierta de un disco. Revela algunas de las gamas de texturas y la gran resolución que se pueden obtener con el *MacPaint*

El potencial artístico del Macintosh se ha plasmado brillantemente en la obra del pintor francés Frédéric Voisin

El *MacPaint*, como todo el software para el Macintosh, está basado en WIMP. Una vez cargado el disco, el escritorio ofrece diversos estilos de dibujo. El usuario puede elegir un pincel de espesores variables, un lápiz, un bote de pintura para rellenar superficies definidas, y un serógrafo para crear un efecto de punteado. Además de estas facilidades de mano alzada, *MacPaint* ofrece la opción de generar líneas rectas, cuadrados, círculos y otras formas geométricas. Cualquier área cerrada se puede rellenar con uno de 38 patrones y sombreados, incluyendo puntos, líneas oblicuas, trabajo de reticulado y cuadros. Los errores se pueden borrar escogiendo el icono de la goma, y una parte de la imagen se puede "enlazar" y trasladar. El menú Goodies (golosinas) ofrece el potencial para otros refinamientos; Fat Bits, por ejemplo, aumenta una superficie en pixels individuales, permitiendo que el usuario suprima o añada diminutos detalles.

Magia y color



Claudia Zeff

La cueva de Aladino
Entrar en el estudio de Frédéric Voisin en los suburbios de París es como entrar en la cueva de Aladino. A primera vista parece muy similar al entorno de trabajo de cualquier otro artista, con paredes y suelos salpicados de pintura, y lienzos sin acabar encaramados sobre caballetes. No obstante, se advierte algo insólito: en una pequeña habitación ajena a toda la pintura y el caos hay un Macintosh sobre un tablero de dibujo. Aquí vemos al artista con un lienzo de 1,5 m de altura. La ilustración se realizó originalmente en el Macintosh, ampliándola después a estas dimensiones. Los bailarines de *break-dance* están pintados con pintura fluorescente y dan la impresión de que se mueven cuando se les aplica luz ultravioleta

Al igual que todo el software Macintosh, *MacPaint* se controla mediante el cursor activado por ratón, que hace innecesario el teclado. Junto con el *MacPaint* también se pueden utilizar accesorios tales como MacTablet, una pequeña tablilla para gráficos con un lápiz óptico.

El artista francés Frédéric Voisin se ha convertido en un entusiasta del *MacPaint*. Hace veinte meses sólo empleaba los útiles tradicionales de su actividad: lápices, lapicero, tinta y papel; ahora utiliza al Macintosh. Su primer contacto con un ordenador lo tuvo con el Apple Lisa de un amigo suyo. Con sólo dibujar una línea con el Lisa comprendió su potencial para generar gráficos. Eso también lo hizo pensar en producir pinturas a gran escala en lugar de ilustraciones. Esta idea la concibió a partir de dos características del Lisa: primero, el hecho de que la imagen en pantalla fuera en blanco y negro y que la forma obvia de añadirle color fuera pintar sobre la salida impresa; segundo, aunque la imagen era pequeña, la gran resolución de la pantalla le sugirió de inmediato las posibilidades de ampliación.

Voisin no perdió tiempo y obtuvo de Apple un Macintosh con el fin de seguir experimentando. Irónicamente, fue un ordenador lo que llevó a Voisin a volcar su interés en la tradición clásica de la pintura. Es decir, hacer un bosquejo sobre el lienzo y pintar sobre el mismo. Pero en vez de utilizar un lápiz para dibujar el boceto, Voisin empieza con

una salida impresa de su ImageWriter, la fotocopia y la amplía hasta tres metros de altura utilizando una máquina copiadora de arquitecto. La siguiente parte del proceso es igualmente tradicional. Voisin pega la copia sobre el lienzo empleandô una cola elaborada con huesos de conejo, mezcla que vienen empleando los artistas desde que comenzaron a pintar sobre lienzo. La cola posee tal adaptabilidad que permite estirar uniformemente un gran trozo de papel sobre un lienzo. Después pinta sobre la copia con pinturas acrílicas fluorescentes para crear vibrantes pinturas posmodernistas de robots, guerreros zulúes y bailarines de *break-dance*. Le gusta exhibir sus lienzos bajo luz ultravioleta, que se combina con la pintura fluorescente para crear un efecto centelleante y pulsátil similar al de una pantalla VDU en color.

En el transcurso del último año Voisin produjo 30 pinturas en su Macintosh de 128 Kbytes utilizando el *MacPaint*, aunque afirma que aún le queda por descubrir el verdadero potencial del programa. Siente un enorme respeto por Bill Atkinson, su diseñador, quien, según cree Voisin, “se ha puesto dentro de la piel de los ilustradores”, anticipándose a sus necesidades. La versión del *MacPaint* que emplea Voisin en realidad es bastante antigua, careciendo de muchos de los refinamientos de las versiones más recientes.



Las líneas de la portada
Ésta es la ilustración original utilizada en la portada de este número de *Mi Computer*. La combinación de la línea compacta y el negro sólido de los rostros con los tonos más suaves del humo ejemplifica la gama de estilos que ofrece *MacPaint*

Arte generado por ordenador

A diferencia de muchos de sus compañeros de profesión, a Voisin no lo intimida la idea de usar un ordenador; lo ve como una herramienta a explorar. Para él, el ratón es como un lápiz: un instrumento que uno utiliza con el cerebro y con la mano. Voisin piensa que finalmente el arte generado por ordenador alcanzará el mismo estatus que el trabajo realizado en un medio más tradicional, como los grabados; la única diferencia residiría en que el original está almacenado en un disco flexible en lugar de estar grabado en una placa de cobre. Voisin cree que esto llevará a una nueva generación de “artistas de masas”, que producirán arte asequible a un gran público en lugar de a unos pocos coleccionistas de arte. El Ministerio de Cultura francés ha puesto a su disposición un Radiance, un ordenador de gran capacidad con una impresora en color. Con el mismo ha producido ediciones limitadas de salidas impresas en color. Voisin ansía probar el Macintosh en color, si bien sólo lo utilizará para ilustraciones y no para producir grandes lienzos.

Voisin aún ha de experimentar con diversos accesorios y software, como el *MacDraw*, disponibles para el Macintosh. Por otra parte, afirma que no podría trabajar sin su Macintosh. “Lo necesito como si se tratara del teléfono o un pincel y pintura. El Macintosh es grandioso. Es una revolución.”

Ediciones “radiantes”
Vemos aquí instantáneas de creaciones de Voisin tomadas del ordenador Radiance. La alta resolución y la facilidad multicolor de la máquina la convierten en un instrumento ideal para crear ilustraciones de calidad. El Ministerio de Cultura francés ha puesto el ordenador a disposición de Voisin para que pueda producir ediciones de salidas impresas en color de su obra



Frédéric Voisin

Cantidad y calidad

En C es posible mejorar y aumentar los tipos de datos definidos por el usuario. Veamos de qué manera

Como lenguaje extensible, el C se puede ampliar para que incluya las propias bibliotecas de definiciones de usted mismo. Ya hemos visto cómo se puede hacer esto con las funciones definidas por el usuario, que simplemente se mantienen en su propio archivo y se accede a ellas utilizando un include en el programa. Asimismo, usted puede aumentar los tipos de datos estándares definidos por el usuario, como int, char, etc. La primera forma en que puede hacerlo es empleando typedef, que permite dar nuevos nombres a tipos estándares. De modo que:

```
typedef int metros;
typedef double vector [10];
typedef char *serie;
```

permitiría declarar variables más adelante en el programa.

Parecería que esto no tiene mayor sentido, pero puede servir para dos cosas. Primero, puede hacer más comprensibles los programas, lo cual, tratándose del C, siempre resulta positivo. En segundo lugar, puede ayudar a superar dificultades para el traslado de programas entre distintas máquinas. La cantidad de bytes que ocupa un int, por ejemplo, no está estandarizada. Quizá diera muchísimo trabajo transportar un programa que utilizara el hecho de que int tuviera cuatro bytes de longitud a una máquina que sólo usara dos bytes.

Es mucho más fácil cambiar un typedef al comienzo del programa en lugar de buscarlo a través de todas las ocurrencias en las que la diferencia fuera significativa. Sería normal querer conservar todos los typedef juntos en un archivo include separado.

El C posee características semejantes al PASCAL en cuanto a la provisión de una estructura de totalización para combinar elementos de datos relacionados de tipos distintos. La instrucción del C struct ofrece una estructura similar a la de RECORD en PASCAL. He aquí un ejemplo de su uso:

```
struct naipe
{
  int número;
  char palo;
}
```

Aquí se define una estructura compuesta por dos partes (un número entero y un palo de caracteres) que van juntas para conformar un naipe de la baraja. Las variables de este tipo se pueden declarar en este y ulteriores puntos del programa. Así:

```
struct naipe baraja [52];
```

formaría una baraja de naipes. La variable se puede declarar junto con la definición de estructura, como en:

```
struct naipe
{
  int número;
  char palo;
}baraja[52];
```

El nombre del rótulo (naipe, en este caso) se puede omitir si se han de declarar todas las variables en un lugar, como en:

```
struct
{
  int número;
  char palo;
}baraja [52], manos [4] [13],* jugadas;
```

pero esto se suele considerar una mala práctica.

Se pueden asignar estructuras completas, pasarlas como argumentos a una función o devolverlas como valores por una función, del mismo modo que cualquiera de las funciones estándares, tales como:

```
naipe [1] [5]=baraja [27];
```

o se puede aludir a los elementos individuales utilizando un punto para una variable normal, y −> para una variable puntero de este tipo, como en:

```
jugadas−>número=baraja [34].número;
```

Estos elementos individuales se comportan como variables comunes del tipo adecuado. Una estructura se puede inicializar del mismo modo que una matriz con el nombre de la variable seguido por una lista de valores para los elementos de la estructura encerrados entre llaves.

Las estructuras reservan palabras de almacenamiento consecutivas para sus elementos; no obstante, unions utiliza el mismo almacenamiento para sus elementos, lo que permite ver la memoria de más de una manera. Por ejemplo, si una máquina usa cuatro bytes para un int, si quisiéramos aludir a estos bytes individualmente, podríamos definir un union que utilizara los mismos cuatro bytes de almacenamiento para un int o cuatro chars:

```
union int_o_char
{
  int parte_int;
  char parte_char [4];
}
```


Estructura flexible
El tipo struct del C permite que el programador combine distintos tipos de datos dentro de una estructura. Por ejemplo, en una única estructura declarada como podemos apreciar en la ilustración se puede representar una baraja de naipes por palo y por número de naipe

Caroline Clayton



A los elementos de un union se alude del mismo modo que a los elementos de una estructura.

En lenguaje normal, se podría aludir a los elementos individuales de una estructura como *campos*; sin embargo, el C utiliza este término para referenciar un tipo determinado de elemento que incluye un tamaño de bit. Ya hemos visto que el C posee operadores a nivel de bit, y los campos nos confieren la facilidad adicional de aludir por nombre a cualquier grupo de uno o más bits en una palabra. Esto se realiza colocando un signo de dos puntos tras el nombre del elemento, seguido por el tamaño de bits. Por ejemplo, en una máquina con una palabra de 16 bits y un tipo int de 16 bits, podríamos definir:

```
structpalabra_en_bytes
{
   unsigned byte0:8,byte1:8;
}
structpalabra_en_bits
{
   unsigned bit0:1, bit1:1, bit2:1, bit3:1, bit4:1,
   bit5:1, bit6:1, bit7:1, bit8:1, bit9:1, bit10:1,
   bit11:1, bit12:1, bit13:1, bit14:1, bit15:1;
}
unión palabra
{
   int w;
   struct palabra_en_bytes w8;
   struct palabra_en_bits w1;
}
```

Esto nos permite aludir a la misma palabra de almacenamiento como un todo, como dos bytes o como 16 bits. Observe el empleo del tipo de datos unsigned (sin signo), que equivale a int con la excepción de que sólo permite valores positivos. Por supuesto, es importante recordar que cuando se especifica de este modo el tamaño, sólo se deben asignar valores en un rango adecuado. A un elemento con un tamaño de un solo bit se le pueden asignar legítimamente los dos valores 0 y 1.

Una mano de bridge

El siguiente ejemplo ilustra algunas funciones que se pueden utilizar para un programa que juegue una mano de bridge. La idea básica del juego (al menos por cuanto nos concierne aquí) es que se mezcle la baraja completa y se reparta entre cuatro jugadores. Una característica de la mano que permite al jugador decidir cómo jugar es el contador de puntos. Éste cuenta cuatro puntos para un As, tres para un Rey, dos para una Dama y uno para un Valet. Aquí las funciones simulan la baraja, el reparto de naipes y los puntos de cuenta. La estructura naipe y la matriz de naipes que componen la baraja se definen e inicializan de modo externo, de manera de poder inicializar la baraja.

```
struct naipe
{
   int número;
   char palo;
}
/* utilizando un espacio mínimo para cada
naipe*/
/* declarar e inicializar baraja*/
struct naipe mano [4] [13], baraja [52]=
{{1,'T'}, {2,'T'}, {3,'T'}, {4,'T'}, {5,'T'},
{6,'T'}, {7,'T'}, {8,'T'}, {9,'T'}, {10,'T'},
{11,'T'}, {12,'T'}, {13,'T'},
{1,'D'}, {2,'D'}, {3,'D'}, {4,'D'}, {5,'D'},
{6,'D'}, {7,'D'}, {8,'D'}, {9,'D'}, {10,'D'},
{11,'D'}, {12,'D'}, {13,'D'},
{1,'C'}, {2,'C'}, {3,'C'}, {4,'C'}, {5,'C'},
{6,'C'}, {7,'C'}, {8,'C'}, {9,'C'}, {10,'C'},
{11,'C'}, {12,'C'}, {13,'C'},
{1,'P'}, {2,'P'}, {3,'P'}, {4,'P'}, {5,'P'},
{6,'P'}, {7,'P'}, {8,'P'}, {9,'P'}, {10,'P'},
{11,'P'}, {12,'P'}, {13,'P'}}
barajar (baraja)
struct naipe baraja [];
/*Asumimos la existencia de un generador de
números aleatorios (random (n)) para devolver
un número aleatorio entre 0 y n-1. Se puede
utilizar la función del sistema rand () que
devuelve un int aleatorio*/
/*Los naipes se barajan intercambiando cada
naipe con algún otro seleccionado al azar*/
{
   int i,j;
   for (i=0;i<52;++i)
   {
      j=random(52);
      swap(&baraja[i],&baraja[j]);
   }
}
swap(p,q)
struct naipe*p, *q;
{
   struct naipe temp;
   temp = *p;
   *p = *q;
   *q= temp;
}
repartir(baraja,mano)
struct naipe baraja[], mano[][];
{
   int i,j,k=0;
   for (i=0;i<4;++i)
      for (j=0;j<13;++j)
         mano [i][j]=baraja[k++];
}
contar_puntos(una_mano)
struct naipe *una_mano;
/* una_mano es un puntero al primer naipe
de un conjunto de 13 en una mano, como
&mano[2][0]*/
{
   int i, contador_puntos=0;
   for (i=0;i<13;++i)
   {
      if(una_mano—> número=1)
         contador_puntos+=4;
      else if (una_mano—> número > 10)
         contador_puntos+=una_
         mano—>número — 10;
         ++una_mano;
/* recuerde que éste ahora apuntará al
siguiente naipe de la baraja*/
   }
   return(contador_puntos);
}
```

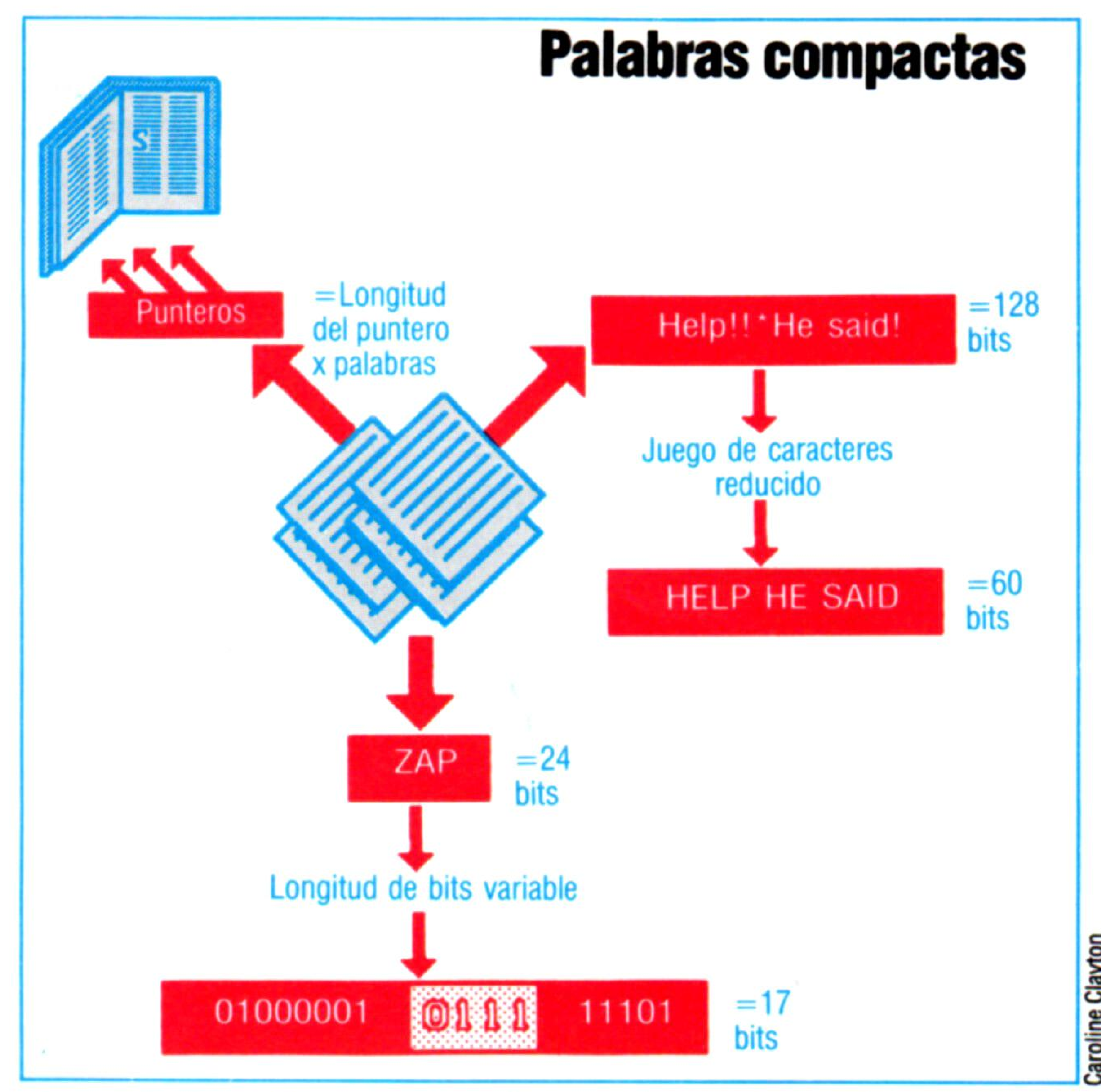
Ahorrar espacio

Iniciamos una serie dedicada a analizar las ventajas de las técnicas de compresión de datos

Tabla de frecuencias

Muchos programas de compresión de textos se basan en las frecuencias de caracteres para una operación eficaz. En inglés, el orden de frecuencia de los caracteres alfabéticos es éste:

ETAONRISHDLFCMUGYPWBVKXJQZ



Encaje justo
La redundancia incorporada del juego de caracteres ASCII con frecuencia requiere alguna forma de compactación de datos, ya sea para maximizar espacio de almacenamiento o bien para reducir los tiempos de transmisión de datos. Existen tres métodos principales de compresión, que normalmente se combinan para conseguir los mejores resultados. Los diccionarios almacenan palabras de uso común, permitiendo construir un archivo de textos como una serie de punteros. Un juego de caracteres reducido elimina los caracteres redundantes (y con frecuencia también las mayúsculas o bien las minúsculas). Por último, la codificación de "bits variables" (o codificación Hoffman) asigna patrones de bits de longitud reducida a las letras que se producen con mayor frecuencia.

En castellano la frecuencia de caracteres difiere del inglés, ordenándose éstos de la siguiente manera: E, A, I, S, U, O, R, N, T, D, L, C, M, Q, P, B, F, G, H, V, Y, J, X, Z, Ñ, K, W

Las técnicas de compresión de textos se pueden utilizar para muchas aplicaciones diferentes, el ejemplo más evidente de las cuales es el de los usuarios de micros personales que deseen programar grandes juegos de aventuras. Pero piense en el ahorro de tiempo y la reducción de las facturas del teléfono que podrían obtener las empresas si comprimieran sus documentos antes de transmitirlos de una a otra oficina. El efecto de las técnicas de compresión eficaces sobre discos flexibles o cartuchos microdrive puede ser igualmente drástico. Es posible conseguir reducciones de volumen de entre el 50 y 60% en archivos de texto moderadamente grandes.

Existen tres técnicas básicas para comprimir archivos. Aquí las describiremos por separado, pero en la práctica suelen utilizarse conjuntamente para obtener la máxima compresión. El primer método, y el más simple, consiste en emplear un juego de caracteres reducido. En la mayoría de los micros, un carácter se almacena en un byte porque éste es el tamaño que le resulta más fácil de manipular al procesador. Esto da ocho bits y, por consiguiente, 256 caracteres posibles. Cuando usted considera que sólo se necesitan 96 caracteres para almacenar los alfabetos incluyendo mayúsculas y minúsculas, números y todos los diversos signos de puntuación del juego ASCII, 256 parece exagerado.

La cantidad mínima de bits enteros necesarios para almacenar 95 caracteres es siete, de modo que codificar la información en siete bits en vez de en ocho reduciría el volumen total en un 12,5%. La solución obvia es emplear menos caracteres. Algunos, como +, *, < y >, por lo general no se utilizan en archivos de documentos y se pueden descartar con razonable seguridad. No obstante, aun el hacer esto no supondrá una diferencia considerable a menos que la aplicación involucrada pueda arreglárselas sin la totalidad del juego de minúsculas o soportar alguna medida igualmente drástica. En consecuencia, el método del juego de caracteres reducido tiene un atractivo limitado. No obstante, se lo emplea para comunicaciones de telex en donde se introduce una dimensión extra. Es decir, utilizar dos caracteres como caracteres de cambio, como la tecla SHIFT de un teclado, para cambiar entre un juego de caracteres alfabéticos en mayúscula y un juego de numerales y signos de puntuación. Ello significa que los caracteres requieren sólo cinco bits cada uno, dando una interesante reducción del 37,5%. En la práctica este porcentaje se reduce por la presencia de caracteres de cambio.

Podemos comprimir el texto agrandando el juego de caracteres. Éste es el segundo procedimiento e implica utilizar caracteres de recambio no empleados por el juego ASCII a modo de "distintivos". Siempre que se encuentre uno de estos valores en el archivo comprimido, se lo utiliza para buscar un valor en una tabla, que contiene una lista de palabras comunes, partes de palabras o frases, tales como "los" o "Uds", o, en documentos financieros, "fiscal", etc. Si estos distintivos se eligen con cuidado, se pueden producir grandes reducciones, si bien al costo de tener que almacenar una tabla de distintivos relativamente larga. La mayoría de los micros, por ejemplo, almacenan programas en BASIC en forma distintivada, con todas las palabras clave almacenadas como bytes individuales.

Se pueden apreciar las ventajas de la entrada distintivada si su máquina le permite colocar una versión no distintivada de un programa en disco o cinta, ya sea directamente (como el Amstrad) o bien LISTando un archivo (como en el Spectrum); la diferencia en tamaño puede resultar sorprendente. Aunque los distintivos para programas en BASIC son, evidentemente, fijos, de hecho los mejores programas distintivadores para grandes documentos buscarán en el archivo a comprimir, hallarán los distintivos óptimos, y almacenarán la tabla de distintivos junto con el texto comprimido.

Este método funciona solamente porque algunas palabras y frases se utilizan con mucha frecuencia, y se puede ampliar para que comprenda caracteres

Compresión ingeniosa

Con el fin de reducir la cantidad de espacio que ocupan los caracteres, podemos codificarlos utilizando una técnica que se conoce como *codificación Hoffman*. El primer paso consiste en calcular la frecuencia relativa de los caracteres implicados; de modo que si quisiéramos comprimir la serie "ASSB", las frecuencias relativas de las tres letras involucradas serían A:1, S:2 y B:1. El siguiente paso es codificar cada carácter, utilizando menos bits para los caracteres más comunes. Luego los códigos se colocan en serie y cuando se requiere codificación el ordenador comienza en el principio del archivo y lo va leyendo, sustituyendo las letras correctas para cada código. Pero se presenta un problema. ¿Cómo sabemos cuándo termina un código y empieza otro? Por ejemplo, si A está codificada como 1, B como 0 y C como 10, ¿cómo podemos decir, cuando detectamos una secuencia de 10, si la misma significa C o AB?

La respuesta reside en codificar cada letra de modo que, leyéndola de izquierda a derecha, podamos estar seguros de que el primer código completo detectado (sea cual fuere su longitud) es el que deseamos. Retomando nuevamente el ejemplo de A, B y C, necesitaríamos codificarlas como 1, 00 y 01 respectivamente.

El programa, que se ejecutará en la mayoría de los micros con ligeras modificaciones o bien ninguna (los usuarios del Spectrum deberán alterar las sentencias de dimensionado de series), hace todo el trabajo pesado por usted. Puede verlo en acción mientras va leyendo los datos proporcionados por defecto, o bien puede proporcionarle los propios caracteres y frecuencias de usted.

Experimente con distintos valores: verá que las reducciones de bits más importantes se producirán cuando las frecuencias relativas muestren grandes fluctuaciones (como en la sentencia de datos proporcionada)

```
10 INPUT "Quieres proporcionar tus propios datos?
   (s/n) ",i$
20 IF i$<> "s" AND i$<>"n" THEN GOTO 10
30 IF i$="n" GOTO 130
40 INPUT "Numero de caracteres a codificar ",n
50 IF N> 255 THEN PRINT "Son demasiados caracteres... 255
   como maximo": GOTO 40
60 DIM c$(n), f(n),r (2*n-1), t(2*n-1)
70 FOR i=1 TO n
80 PRINT "Entra numero de caracter ";i: INPUT s$
90 IF LEN(s$)> 1 THEN PRINT "Solo un caracter por
   vez...":GOTO 80
100 PRINT "Entra la frecuencia de numero de caracter ";i:INPUT
   f(i)
110 a$=a$+s$
120 NEXT i:GOTO 150
130 n=26: DIM c$(n), f(n), r(2*n-1), t(2*n-1)
140 FOR i=1 TO n:READ s$,f(i):a$=a$+s$:NEXT i
150 FOR i=1 TO n:r(i)=f(i): NEXT i
160 FOR i=n+1 TO 2* n-1
170 z=9999:v=9999:k=0:g=0
180 FOR q=1 TO i-1
190 IF t(q)<>0 THEN GOTO 220
200 IF r(q)< z THEN g=k:v=z:k=q:z=r(q):GOTO
   220
210 IF r(q)< v THEN g=q:v=r(q)
220 NEXT q
230 p=i
240 r(p)=z+v: t(k)=-p: t(g)=p
250 NEXT i
260 FOR i=1 TO n
270 c$(i)=""
280 p=i
290 IF t(p)=0 THEN GOTO 340
300 IF t(p)> 0 THEN c$(i)="0"+c$(i): GOTO 320
310 c$(i)="1"+c$(i)
320 p=ABS(t(p))
330 GOTO 290
340 NEXT i
350 FOR i=1 TO n
360 PRINT MID$(a$,i,l),r(i),c$(i)
370 NEXT i
380 END
390 DATA e, 250, t, 220, a, 24, o, 23, n, 22, r, 21
400 DATA i, 20, s, 19, h, 18, d, 17, l, 16, f, 15
410 DATA c, 14, m, 13, u, 12, g, 11, y, 10, p, 9
420 DATA w, 8, b, 7, v, 6, k, 5, x, 4, j, 3, q, 2, z, 1
```

individuales. Este enfoque constituye la base de la tercera técnica de compresión. En los dos procedimientos anteriores, la cantidad de bits utilizada para cada carácter o distintivo era fija. No obstante, la compresión por longitud de bits variables (también conocida como *codificación Hoffman*), como su nombre implica requiere codificar los caracteres más comunes en menos bits que los caracteres menos comunes.

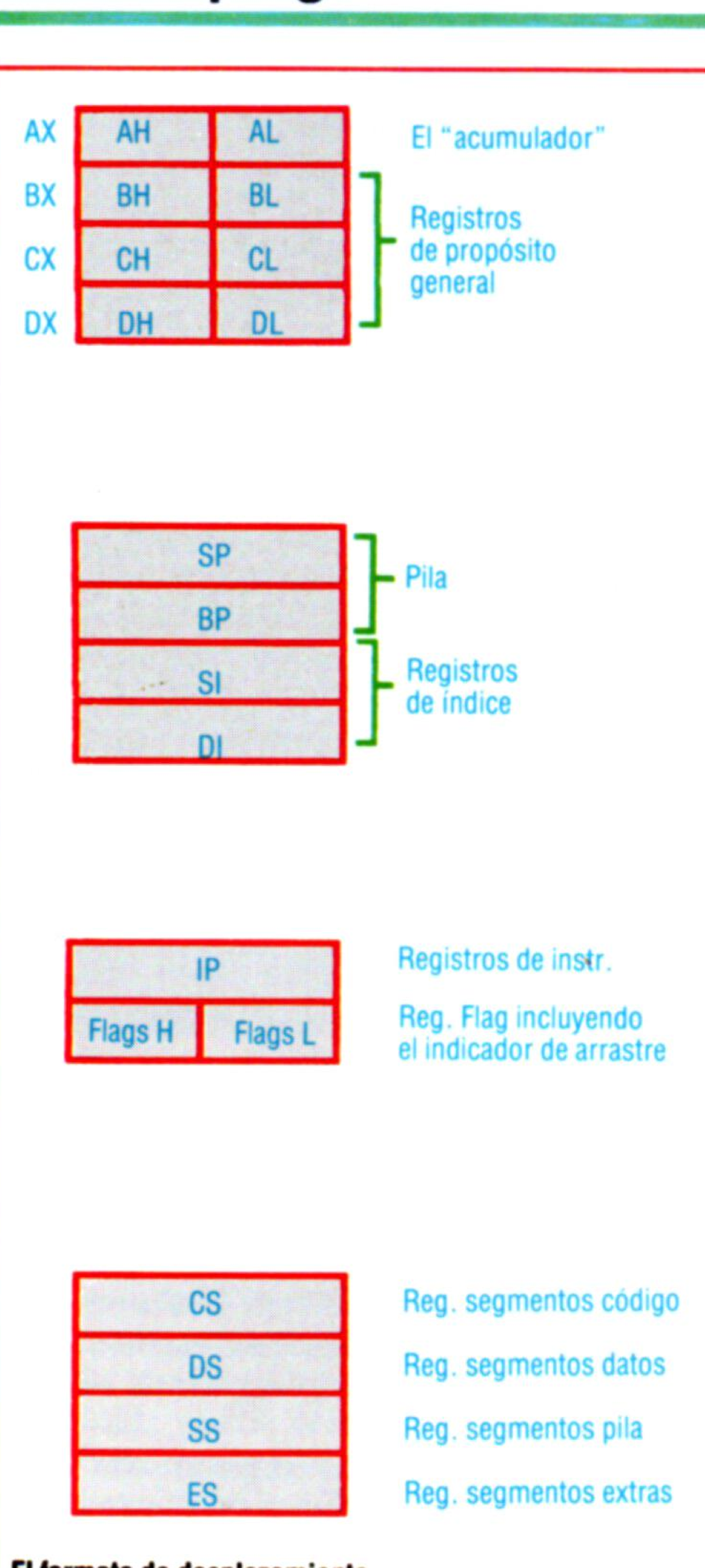
Este enfoque no es nuevo: Samuel Morse lo utilizó, por ejemplo, para su código telegráfico de "puntos y rayas". Sin embargo, Morse tuvo la ventaja de contar con "caracteres" extras en la forma de vacíos en la transmisión, que utilizó para delimitar caracteres y palabras. Aun sin disfrutar de esta ventaja, Hoffman aportó un sistema que se basaba en el hecho de que ninguno de sus caracteres se podía componer colocando accidentalmente otros dos caracteres distintos consecutivos.

Sin embargo, esta restricción significa que la mayoría de los caracteres poco comunes pueden requerir el almacenamiento de 17 bits o más. Con el fin de reducir la cantidad máxima de bits, y para añadir la capacidad de incluir también algunos distintivos, se ha desarrollado un método refinado que (aunque de longitud variable) utiliza solamente secuencias de palabras de cuatro bits. Una palabra de cuatro bits se puede usar para almacenar 16 patrones diferentes, de los cuales tres se emplean como patrones de señal para dar información sobre los cuatro elementos de bits siguientes. Los otros 13 se utilizan para los 11 caracteres más usados, junto con el carácter de espacio y el carácter de salto de línea.

Los tres valores de señal se emplean para denotar ya sea que la siguiente palabra de cuatro bits se ha de utilizar como un valor de carácter de una de las 16 palabras más comunes, o bien que los ocho bits siguientes se deben utilizar para denotar uno de los caracteres menos usados o una palabra menos común. Este procedimiento permite el uso de los 96 caracteres completos, más un juego de 205 distintivos para palabras comunes. Se podrían incluir otros refinamientos en el supuesto de que todas las palabras del diccionario comenzaran con un espacio (a menos que se las coloque al comienzo de una línea), ahorrando de este modo cuatro bits cada vez que se utilice una de ellas. Estas características en combinación dan excelentes resultados, con una reducción que suele superar el 50%.

Llamadas a función

Concluiremos esta breve serie dedicada a estudiar el MS-DOS analizando este sistema operativo desde el punto de vista del programador



El formato de desplazamiento
El procesador 8086/8088 puede direccionar hasta un megabyte de memoria. Esto requiere direcciones de 20 bits, y puesto que el 8088 se basa en registros de 16 bits, por consiguiente se necesitan bits extras para conformar una dirección. Los diseñadores del 8086/8088 sortearon este problema adoptando un segmento: formato de desplazamiento. El registro de segmentos direcciona un bloque de memoria de 64 Kbytes y el registro de desplazamiento identifica al byte dentro del bloque. Por ejemplo, la dirección de la instrucción actual está retenida en CS:IP. Los registros más importantes son A, B, C, D y el registro Flag. Se puede aludir a estos registros como una palabra de 16 bits completa utilizando el sufijo 'X' (por ejemplo, AX) o como bytes *low* y *high* utilizando los sufijos 'L' y 'H'. El byte *low* del registro D es, en consecuencia, DL

Caroline Clayton

Las muchas llamadas al sistema disponibles en MS-DOS por lo general se clasifican en seis categorías diferentes:

- E/S de caracteres desde y hacia los dispositivos estándares del sistema.
- Tratamiento de archivos (incluyendo la gestión del directorio).
- Gestión de memoria.
- Gestión de procesos.
- Llamadas misceláneas a "funciones" del sistema.
- Llamadas especiales a Microsoft Networks.

Todos estos servicios del sistema se pueden llamar desde cualquier aplicación, y proporcionan una interface unificada y exhaustiva que asegura la compatibilidad entre distintas versiones del OS. Las llamadas a direcciones absolutas o directas a dispositivos de hardware no serán necesariamente portables a otros sistemas MS-DOS o, incluso, compatibles con versiones DOS diferentes, ni siquiera siendo del mismo fabricante.

El empleo de la palabra *función* para describir todas las llamadas al sistema es terminología MS-DOS estándar, derivada del BCPL y del C a través del Unix. *No* significa que el servicio no sea más que una función que devuelva algún dato. Asimismo, incluye rutinas que verdaderamente hacen algo útil (como abrir archivos, etc.). Éstos, por supuesto, son procedimientos, pero en toda la literatura son funciones denominadas de una forma vaga.

Las funciones de fines generales más útiles son las llamadas *misceláneas*. Cada una se ve afectada por una interrupción de software y siempre implica la consecución de los siguientes pasos:

1. Trasladar los datos requeridos a los registros adecuados (en DL, p. ej., se podría requerir un número de unidad).
2. Colocar el número de función en AH (el byte *high* de AX).
3. Generar la interrupción de software 21H.

Todo dato devuelto por la función se hallará en los registros 8086 al retornar de la llamada, ya sea directamente o a través de un puntero o dirección. Por lo tanto, la función DOS 19H, por ejemplo (todos los números, de función son hexadecimales), devuelve la unidad de disco seleccionada actualmente como un código en el registro AL. No requiere ningún código de acción, de modo que se podría codificar como:

```
mov   ah,19H   ;tomar unidad seleccionada
int   21H      ;llamar a la función
```

El resultado se devuelve en el registro AL, y es un número que representa a la unidad (A=0, B=1, y así sucesivamente). Sumando a este resultado el código ASCII para el carácter 'A' y llamando a la función DOS 05H (imprimir el código de DL como un carácter ASCII), podríamos visualizar la unidad conectada como un carácter en mayúsculas:

```
add al, 'A'        ;convertir 0 en 'A', 1 en 'B', etc.
mov dl, al         ;trasladarlo al registro DL
mov ah,05H         ;imprimir carácter
int 21H            ;llamar a la función
```

Los códigos de instrucción empleados en el ejemplo anterior son para el propio ensamblador de Microsoft, pero no siempre el OEM se lo facilita al usuario final. Afortunadamente, no es esencial tener un ensamblador; como veremos, por lo general es muy fácil llamar al MS-DOS desde lenguajes de alto nivel. Sin embargo, si usted tiene intenciones de hacer muchísima programación en código máquina, asegúrese de que adquiere un ensamblador que genere archivos en código máquina reubicables en formato Intel (.OBJ) estándar. Éste es el estándar *de facto* para máquinas de la familia 8086, y la mayoría de los montadores (incluyendo aquellos para PASCAL, FORTRAN, C, etc.) de proveedores afamados le permitirán enlazar módulos de edición escritos en cualquiera de estos lenguajes (con bibliotecas de código máquina ensambladas si así lo deseara).

Una función completa

Finalmente, he aquí una llamada a función de alto nivel completa. Muy a menudo es necesario averiguar cuánto espacio de disco hay disponible antes de escribir datos en un archivo. Llamando a la función del DOS 36H (decimal 54) podemos comprobar el disco antes de escribir en él y, de ser necesario, darnos la opción de cerrar archivos o intercambiar discos si así se requiriera. Para ejecutar esta llamada al sistema, se coloca en AH el número de función del modo usual, y DL debe contener el número de unidad; la unidad A es 1, la B es 2, etc. Un código de 0 indica la unidad por defecto. Luego es preciso inicializar el campo DL (asociado al byte *low* del registro D) en el valor requerido, colocar 36H en AH y realizar la llamada al sistema. Al retornar, AX contendrá el código de error OFFFFH si algo no marcha bien (que se haya dado un código de unidad no válido, p. ej.); de lo contrario, se devuelven los siguientes datos (mostrando lo que retiene cada registro):

AX Cantidad de sectores por grupo (*cluster*)
BX Grupos disponibles
CX Cantidad de bytes por sector
DX Cantidad total de grupos

De modo que el espacio total de disco disponible (en bytes) es el producto de los valores:

grupos × sectores por grupo × bytes por sector

Observe que un grupo es una unidad de asignación de espacio de disco, y corresponde a una cantidad de sectores completos.

```
FUNCTION EspacioDisco (unidad:integer):
integer;
  {devuelve la cantidad de bytes libres en la
  unidad}
  VAR
     registros: SysReg;{ver manual}
  BEGIN
    WITH registros DO
      BEGIN
        DL:=unidad;{0=defecto,1=A,etc.}
        AH:=36H; {número de función}
        sistema (registros);{llamarlo}
        IF AX=0FFFFH
          THEN {hubo un error, de modo que:}
            BEGIN
            WriteLn ('EspacioDisco: ERROR
            (código no válido?)');
            EspacioDisco:=0{por lo que
            sabemos!}
          END
        ELSE
            EspacioDisco:=AX*BX*CX
      END {sectores*grupos*bytes}
END: {EspacioDisco}
```

Programación de alto nivel

Una alternativa al empleo de un ensamblador es el sustituto del BASIC de colocar (POKE) instrucciones (ensambladas manualmente) en un bloque conveniente de memoria libre. Esto, no obstante, sólo se recomienda para tareas bastante triviales tales como en las cortas rutinas que acabamos de ilustrar. Usted necesitará una lista de *opcodes* Intel, un detallado mapa de memoria de su máquina, conocimiento sobre el espacio para programas transitorios, ¡y una buena dosis de paciencia! Especialmente desde el punto de vista de la legibilidad, lo mejor es evitar esta opción. Afortunadamente, muchas implementaciones de lenguajes compilados para el MS-DOS incluyen las ampliaciones necesarias para escribir código máquina externo y enlazarlo con el programa principal.

El propio PASCAL-86 de Microsoft posee un procedimiento denominado DOSXQQ, que da solicitudes de "función" directas. El MT+86 de Digital Research posee una rutina similar. Quizá el mejor enfoque (y, por cierto, el más sencillo) sea el adoptado tanto por el compilador homologado por ISO de Prospero Software (Pro PASCAL), muy bien considerado, como el económico Turbo PASCAL. Éstos ofrecen un vehículo excelente para programación del DOS sin necesidad de utilizar lenguaje ensamblador.

Tanto el Pro PASCAL como el Turbo PASCAL poseen un descriptor TYPE que mapea todos los registros 8086 esenciales en un RECORD (registro), cuyos campos se pueden inicializar antes de una llamada y acceder luego a ellos para devolver datos desde la función DOS. La sintaxis es similar en cada caso (para detalles completos, remitirse al manual de PASCAL).

Prospero posee un procedimiento llamado, como es lógico, system, que toma un parámetro de variable del tipo mencionado (SysReg) y realiza la auténtica llamada a la función. De modo que el procedimiento completo "imprimir la unidad actualmente seleccionada" que acabamos de dar en ensamblador se convertiría en:

```
WITH registros DO
  BEGIN {imprimir la unidad seleccionada:}
    AH:=25; {código de función 19H}
    system (registros);{llamarla}
    WriteLn ('La unidad seleccionada actualmente
      es', chr (AL+ord ('A')),':')
  END
```

Una pila de ideas

Nuestro examen se centra en las relaciones entre la subrutina y la pila, como paso previo al estudio de los parámetros

Comenzaremos analizando un ejemplo que realiza dos llamadas a la subrutina CALC, ya presentada anteriormente.

```
                  ....            *líneas de código
2000 4EB8         JSR CALC        *llamada a la
                                   subrutina
                  (...líneas de código...)
2100 4EB8         JSR CALC        *otra vez
2102 4000
                  (...líneas de código...)
4000 3200    CALC MOVE D0,D7      *entrada subrutina
                  (...líneas de código...)
4100 4E75         RTS             *retorno a llamada
```

Cada vez que se ejecuta la instrucción JSR, el 68000 coloca el contenido del contador de programa (la dirección de la instrucción siguiente a la instrucción JSR) en la pila y carga la dirección de la rutina especificada por JSR en el PC. La subrutina se ejecuta seguidamente y cuando encuentra RTS el 68000 saca la dirección de retorno (conocida como *dirección de enlace*) de la pila y la pone en el PC. Es de observar que la dirección de enlace (*linkage*) se guarda en formato de palabra larga completa, por lo que en el ejemplo, tras la primera ejecución de CALC, la pila y el PC contendrán los valores que se ilustran en el dibujo (página contigua).

Después de la ejecución de RTS, al final de la subrutina CALC, se restaura la dirección de enlace en el PC y la pila queda como se ilustra en la segunda parte del dibujo. Es de observar que la dirección de enlace continúa escrita en la pila, y será sobreescrita la siguiente vez que ésta sea usada.

Este mecanismo de enlace es obra del hardware del 68000. Pero su significado es que si anidamos subrutinas, el mecanismo de enlace también se cuidará automáticamente de esta situación.

Es claro que este método de apilar las direcciones de enlace automáticamente se cuida de las subrutinas anidadas mediante la ampliación de la pila. Supongamos que CALC se llama a sí misma (llamada recursiva). De nuevo el mecanismo de enlace se encargará de la situación apilando todas las direcciones de enlace una encima de la otra hasta que acaben las llamadas recursivas, o hasta que no se venga abajo el funcionamiento por culpa de alguna violación de dirección, es decir, ¡porque la pila ha crecido en exceso!

Volvamos ahora al problema de cómo se pasan los parámetros, tanto de entrada como de salida. En el capítulo anterior pasamos datos a la subrutina CALC mediante D1 y recuperamos datos de ella con D2. Esta sencilla solución del problema basta para pequeños programas, pero precisamos una solución más general para programas mayores, sobre todo pensando en que sólo se dispone de unos cuantos registros.

Un método utilizado para superar esta dificultad, en especial con compiladores, es el empleo de la pila como instrumento para pasar parámetros. Así, escribiríamos esto para pasar parámetros a CALC:

```
2000  MOVE PARAM1,-(SP)  *pone el primer
                          parámetro en la pila
2004  MOVE PARAM2,-(SP)  *y el segundo
2008  JSR CALC           *llamada a CALC
200A  ......
```

donde la pila contendría, al entrar en CALC:

```
              PARAM1
              PARAM2
              enlace ls
SP en CALC→   enlace ms
```

Para acceder a PARAM1 tendremos que utilizar un desplazamiento en el SP de modo que salte por encima de la dirección de enlace:

```
ADDQ  #6,SP    *ajuste puntero
MOVE  (SP),D4  *toma el
                parámetro
```

o, más sencillamente:

```
MOVE  6(SP),D4
```

Naturalmente, si alteramos el puntero de la pila, del modo que sea, debemos cerciorarnos de que antes de ejecutar la instrucción RTS apunte a la dirección de enlace, de otro modo pueden sobrevenir resultados desastrosos e impredecibles. Este principio es también válido cuando CALC devuelve parámetros mediante la pila, a menos que no sobreescriba uno o los dos parámetros de entrada, como es obvio.

No obstante, un método más sencillo será el empleo de una pila independiente para los parámetros, por ejemplo, la A6, dejando tranquilo al puntero de la pila (que se reserva para las funciones del hardware). En este caso, la llamada sería:

```
MOVE  PARAM1,-(A6)     *pone primero
                        el parámetro en la
                        pila A6
MOVE  PARAM2,-(A6)     *y el segundo
JSR   CALC             *guarda el SP
                        para el enlace
```

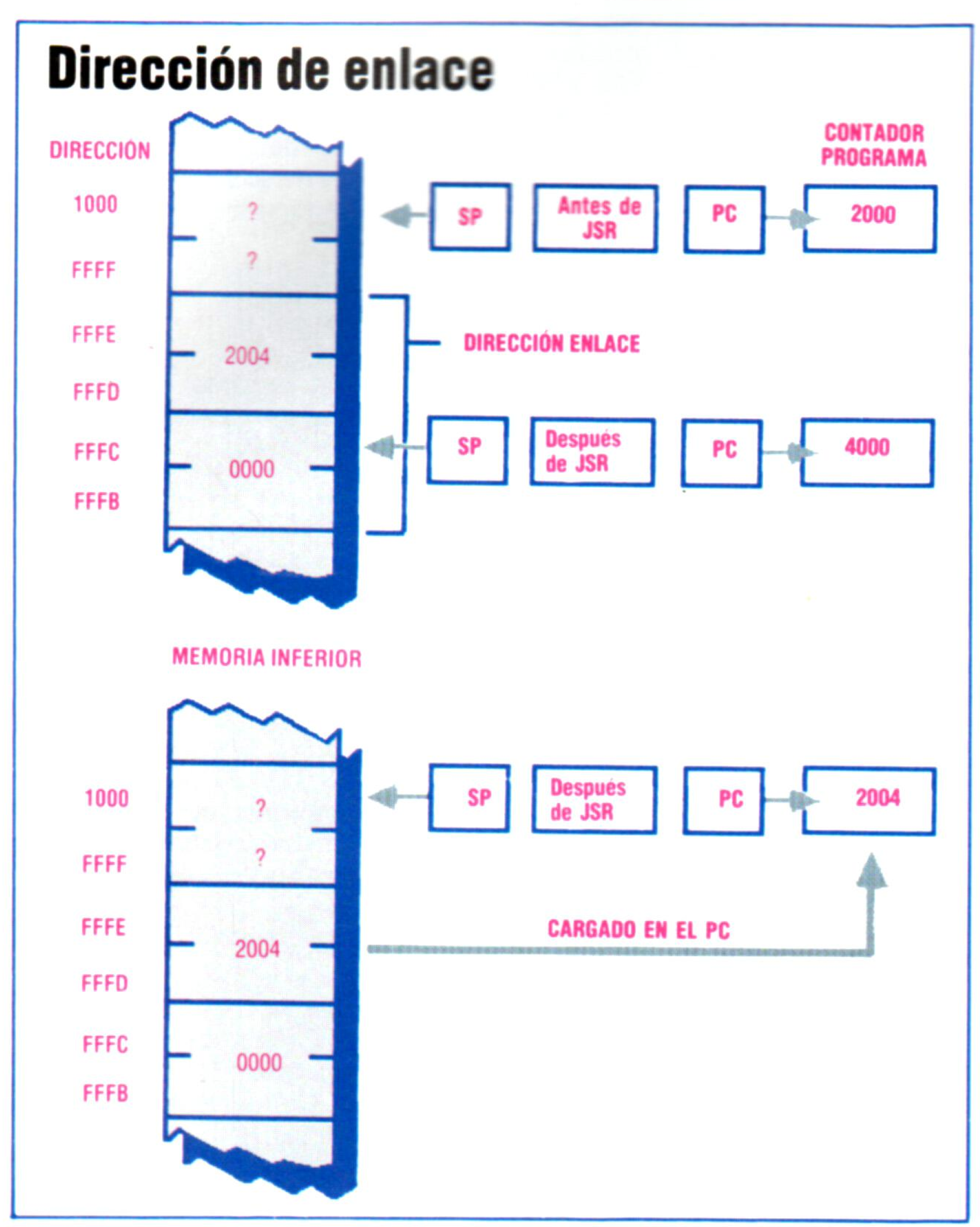
y dentro de la subrutina podemos fácilmente tomar los parámetros empleando, por ejemplo:

```
MOVE  (A6)+,D2         *toma el segundo
                        parámetro
MOVE  (A6)+,D1         *y el primero
```

ya que no existe el peligro de "pisar encima" de algún enlace de llamada a rutina. Es de notar que se toman los parámetros en el orden inverso respecto a la llamada.

Las pilas se emplean en los modernos lenguajes de alto nivel estructurados en bloques, como el PASCAL, en una forma muy ordenada y estructurada.





Vínculo permanente
El 68000 guarda una dirección de retorno (la *dirección de enlace*) en la pila antes de pasar el control del programa a la subrutina. Esta dirección se guarda en formato de palabra larga en la pila y se restaura en el contador de programa (*program counter*: PC) cuando se encuentra una instrucción RTS

La estructura de pila típica permite parámetros y variables locales de un procedimiento (que se implementará como subrutina en el código a ejecutar) que ha de permitir espacio en la pila dentro de un área definida. Así, por ejemplo:

```
Procedure Calcular (xval, yval:int);
var
        store:int;
begin
        store:=2*xval+yval^2;
```

como un fragmento de PASCAL tendría los parámetros de entrada xval e yval, y la variable local store, espacio reservado en la pila. Además, todo almacenamiento temporal sin nombre que necesite el compilador (pongamos por caso, para retener el componente yval^2 de store mientras se evalúa 2*xval) empleará la pila en su manera normal de poner y sacar.

Este arreglo está bien estructurado desde la perspectiva del compilador, pero en algunos ordenadores puede llevar a un nada despreciable dispendio en la manipulación de los datos de pila. En el 68000, la situación está facilitada considerablemente por el uso de dos instrucciones: LNK (*link*: enlazar) y UNLNK (*unlink*: desenlazar).

Estas instrucciones se emplean juntas y permiten la fácil manipulación de datos mediante la reserva de bloques de memoria dentro de la pila para uso de la subrutina. Después de una entrada a la subrutina, LNK establece un registro de direcciones definido (denominado *frame pointer*, FP, puntero marco) en un área de datos de la pila, y baja el SP de la pila un cierto número de posiciones. Por ejemplo, si la pila era:

```
        param1
        param2
        enlace ls
SP→     enlace ms
```

después de ejecutar JSR, el estado, tras la ejecución de la instrucción LNK, sería:

```
            param1
            param2
            enlace ls
            enlace ms
FP →        FP antiguo
|           denominado
desplazamiento local
|           variables
SP
```

El espacio de la pila crecería "hacia abajo", como indica el dibujo, dado que la subrutina necesita mayor espacio de trabajo.

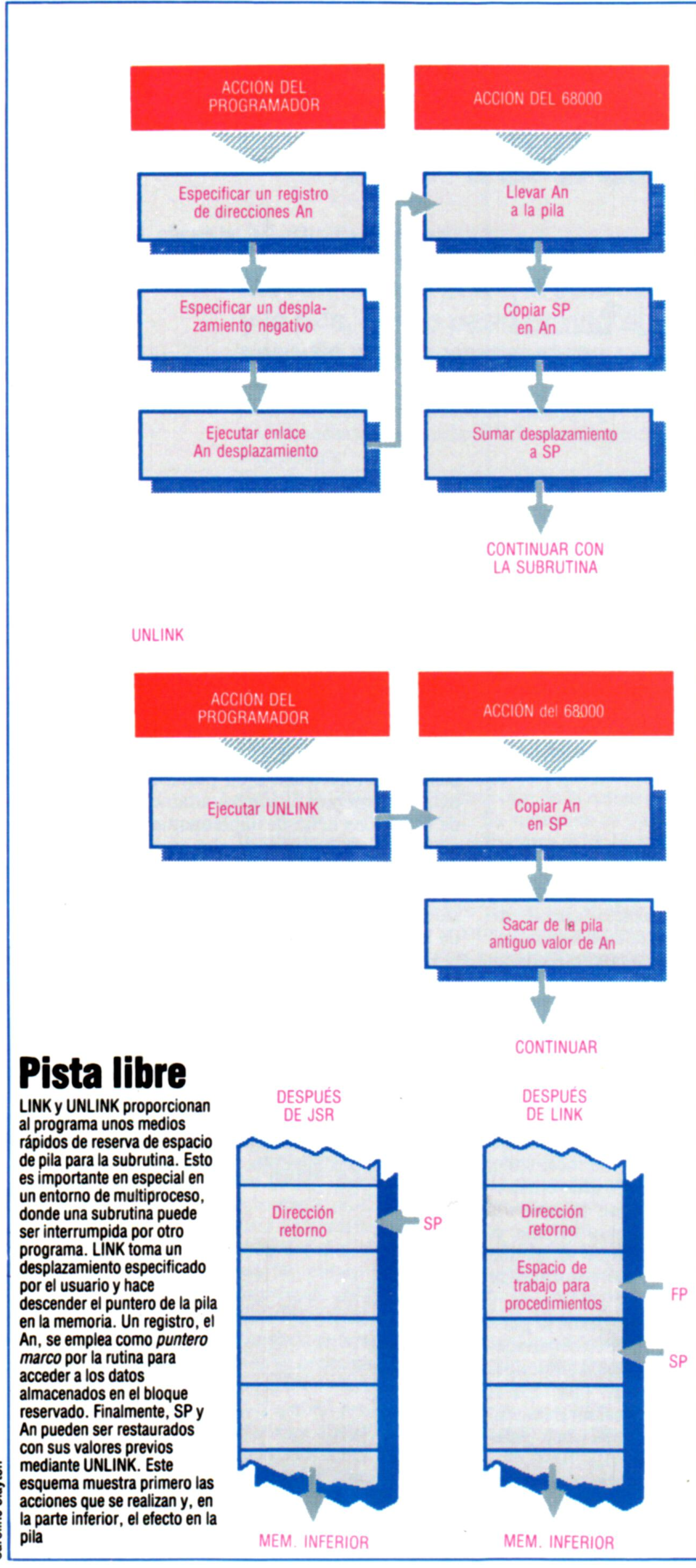
Una vez ejecutada la instrucción UNLNK al final de la subrutina, inmediatamente antes de la instrucción RTS, los punteros volverán a su estado previo a la entrada en la subrutina.

Hasta aquí hemos visto los dos extremos respecto del pase de parámetros en el 68000. En un extremo tenemos el empleo sencillo de los registros en el que el parámetro se carga en el registro de los datos (en realidad, un registro de direcciones si deseamos pasar una referencia a un parámetro). En el otro extremo tenemos el empleo altamente estructurado de la pila con el fin de implementar el pase de parámetros en un lenguaje de alto nivel. Veamos ahora un término medio, un estado en que podemos hacer un empleo algo más sofisticado que la utilización del registro sencillo pero que no implica el uso de la pila.

El primer método consiste en emplear un área definida de datos dentro de un grupo de subrutinas (p. ej., llamado módulo) que contiene todos los parámetros de entrada/salida asociados con ese módulo. Esta área de datos no es global, pero sólo es empleada por un conjunto definido de subrutinas para el propósito específico de pasar parámetros. Cada subrutina sabrá exactamente dónde tomar los parámetros y dónde dejar su parámetro de salida al final. Así, por ejemplo, podríamos escribir:

```
CALC  LEA   INPUTS,A4     *apunta al
                           área parám. entrada
      MOVE  (A4),D0       *toma el primer
                           parámetro
      MOVE  (A4)+,D1      *y también el
                           segundo
      ....                *instrucciones
                           diversas
      MOVE  D5,OUTPUT     *entra el
                           parám. salida
      RTS
```

Naturalmente, podemos pasar un puntero al área de parámetros en un registro de direcciones definido. Esto significaria que la primera línea en el ejemplo anterior resulta innecesaria. En efecto,



Pista libre
LINK y UNLINK proporcionan al programa unos medios rápidos de reserva de espacio de pila para la subrutina. Esto es importante en especial en un entorno de multiproceso, donde una subrutina puede ser interrumpida por otro programa. LINK toma un desplazamiento especificado por el usuario y hace descender el puntero de la pila en la memoria. Un registro, el An, se emplea como *puntero marco* por la rutina para acceder a los datos almacenados en el bloque reservado. Finalmente, SP y An pueden ser restaurados con sus valores previos mediante UNLINK. Este esquema muestra primero las acciones que se realizan y, en la parte inferior, el efecto en la pila

Caroline Clayton

este método se basa en la creación de áreas independientes de pilas para los parámetros de ENTRADAS y SALIDA.

El segundo método vuelve al procedimiento más sencillo de todos, el del uso directo de los registros, pero se aprovecha de la instrucción MOVEM. Examinemos primero esta instrucción. Sea la siguiente formulación:

```
MOVEM lista registro,AREARESERVA
```

Esto guardaría los registros definidos en la lista de registros en la dirección absoluta AREARESERVA y siguientes posiciones hacia arriba.

Por ejemplo:

```
MOVE D3/D5/A2, AREARESERVA
```

cargará D3 en AREARESERVA, D5 en AREARESERVA+2, y A2 en AREARESERVA+4. Podríamos guardar, asimismo, los registros en la pila con:

```
MOVEM D3/D5/A2,–(SP)
```

que apilará el registro en direcciones consecutivas decrecientes. Si los registros consecutivos se han de guardar, entonces el ensamblador acepta la versión taquigráfica:

```
MOVEM D1–D5/A4,–(SP)
```

donde se apilarán desde D1 hasta D5 y A4.

Podemos también almacenar los registros empleando la lista de registros como destino en la instrucción MOVEM. Por ejemplo:

```
MOVEM–(SP),D1–D5/A4
```

restaurará los registros almacenados en el ejemplo anterior.

Debemos preguntarnos para qué nos sirve esto en el pase de parámetros. La respuesta está en la conveniencia de almacenar registros que se han distribuido para un objetivo específico, por ejemplo para uso del sistema y después quedar libres para usar los registros en lo que queramos.

Por ejemplo, si reservamos D0 y D1 como registros de parámetros, entonces con tal de que almacenemos los restantes registros en la entrada a la subrutina, podríamos utilizar los restantes registros según se exija en la subrutina.

Por ejemplo:

```
CALC   MOVEM D2–D7,–(SP)   *guarda antiguos
                            D2-D7
       MOVE    D0,D2       *y carga el
                            primer parámetro
       ....                *instrucciones
                            que emplean
                            D2-D7
       MOVEM –(SP),D2–D7   *y restaura
                            valores antiguos
       RTS
```

De esto se puede colegir que el 68000 nos permite el empleo de varios métodos para pasar parámetros a las subrutinas y ¡nos proporciona además algunas herramientas útiles para hacerlo!

Editorial Delta, S.A.



10116
9 788485 822836